Please type a plus sign (+) inside this box [+]

PTO/SB/05 (12/97)
Approved for use through 09/30/00. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

UTILITY PATENT APPLICATION TRANSMITTAL

(Only for new nonprovisional applications under 37 CFR 1.53(b))

Attorney Docket No. 003057.P003DC3 Total Pages 6

First Named Inventor or Application Identifier Faan-Hoan Liu

Express Mail Label No. EM088414194US

ADDRESS TO: **Assistant Commissioner for Patents**
Box Patent Application
Washington, D. C. 20231

APPLICATION ELEMENTS
See MPEP chapter 600 concerning utility patent application contents.

1. x Fee Transmittal Form
(Submit an original, and a duplicate for fee processing)

2. x Specification (Total Pages 39)
(preferred arrangement set forth below)
- Descriptive Title of the Invention
- Cross References to Related Applications
- Statement Regarding Fed sponsored R & D
- Reference to Microfiche Appendix
- Background of the Invention
- Brief Summary of the Invention
- Brief Description of the Drawings (if filed)
- Detailed Description
- Claims
- Abstract of the Disclosure

3. x Drawings(s) (35 USC 113) (Total Sheets 17)

4. x Oath or Declaration (Total Pages 2)

a. ___ Newly Executed (Original or Copy)

b. x Copy from a Prior Application (37 CFR 1.63(d))
(for Continuation/Divisional with Box 17 completed) **(Note Box 5 below)**

i. ___ DELETIONS OF INVENTOR(S) Signed statement attached deleting inventor(s) named in the prior application, see 37 CFR 1.63(d)(2) and 1.33(b).

5. x Incorporation By Reference (useable if Box 4b is checked)
The entire disclosure of the prior application, from which a copy of the oath or declaration is supplied under Box 4b, is considered as being part of the disclosure of the accompanying application and is hereby incorporated by reference therein.

6. _ Microfiche Computer Program (Appendix)

7. __ Nucleotide and/or Amino Acid Sequence Submission
(if applicable, all necessary)
a. ____ Computer Readable Copy
b. ____ Paper Copy (identical to computer copy)
c. ____ Statement verifying identity of above copies

ACCOMPANYING APPLICATION PARTS

8. _____ Assignment Papers (cover sheet & documents(s))
9. _____ a. 37 CFR 3.73(b) Statement (where there is an assignee)
_____ b. Power of Attorney
10. _____ English Translation Document (if applicable)
11. _____ a. Information Disclosure Statement (IDS)/PTO-1449
_____ b. Copies of IDS Citations
12. x Preliminary Amendment
13. x Return Receipt Postcard (MPEP 503) (Should be specifically itemized)
14. _____ a. Small Entity Statement(s)
b. Statement filed in prior application, Status still proper and desired
15. _____ Certified Copy of Priority Document(s) (if foreign priority is claimed)
16. _____ Other: _____

17. **If a CONTINUING APPLICATION,** check appropriate box and supply the requisite information:
X Continuation ____ Divisional ____ Continuation-in-part (CIP)
of prior application No: : 08/484,949; which is a Divisional of U.S. Patent No. 5,446,877; which is a Continuation of 08/105,478; which is a Continuation of 07/612,540.

18. **Correspondence Address**
_____ Customer Number or Bar Code Label _____
(Insert Customer No. or Attach Bar Code Label here)
or
X Correspondence Address Below

NAME Daniel M. De Vos
BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP
ADDRESS 12400 Wilshire Boulevard
Seventh Floor
CITY Los Angeles STATE California ZIP CODE 90025-1026
Country U.S.A. TELEPHONE (408) 720-8598 FAX (408) 720-9397

PTO/SB/17(10/96)
Approved for use through 09/30/98. OMB 0651-0032
Patent and Trademark Office: U.S. DEPARTMENT OF COMMERCE
Under the Paperwork Reduction Act of 1995, no persons are required to respond to a collection of information unless it displays a valid OMB control number.

FEE TRANSMITTAL

TOTAL AMOUNT OF PAYMENT ($) 790.00

Complete if Known:
Application No. To be assigned
Filing Date 08-21-98
First Named Inventor Faan-Hoan Liu
Group Art Unit To be assigned
Examiner Name To be assigned
Attorney Docket No. 003057.P003DC3

METHOD OF PAYMENT (check one)

1. [X] **The Commissioner is hereby authorized to charge indicated fees and credit any over payments to:**

   Deposit Account Number 02-2666
   Deposit Account Name _____

   [ ] **Charge Any Additional Fee Required Under 37 CFR 1.16 and 1.17**

   [ ] **Charge the Issue Fee Set in 37 CFR 1.18 at the Mailing of the Notice of Allowance, 37 CFR 1.131(b)**

2. X **Payment Enclosed**
   X **Check**
   _____ **Money Order**
   _____ **Other**

FEE CALCULATION (fees effective 10/01/97)

1. FILING FEE

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
101	790	201	395	Utility application filing fee	790.00
106	330	206	165	Design application filing fee	
107	540	207	270	Plant filing fee	
108	790	208	395	Reissue filing fee	
114	150	214	75	Provisional application filing fee	

SUBTOTAL (1) $ 790.00

2. CLAIMS

			Extra		Fee from below		Fee Paid
Total Claims	1	– 20 =	0	X		=	
Independent Claims	1	– 3 =	0	X		=	
Multiple Dependent Claims				X		=	

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
103	22	203	11	Claims in excess of twenty	
102	82	202	41	Independent claims in excess of 3	
104	270	204	135	Multiple dependent claim	
109	82	209	41	Reissue independent claims over original patent	
110	22	210	11	Reissue claims in excess of 20 and over original patent	

SUBTOTAL (2) $0

FEE CALCULATION (continued)

3. ADDITIONAL FEES

Large Entity Fee Code	Large Entity Fee ($)	Small Entity Fee Code	Small Entity Fee ($)	Fee Description	Fee Paid
105	130	205	65	Surcharge - late filing fee or oath	
127	50	227	25	Surcharge - late provisional filing fee or cover sheet	
139	130	139	130	Non-English specification	
147	2,520	147	2,520	For filing a request for reexamination	
112	920*	112	920*	Requesting publication of SIR prior to Examiner action	
113	1,840*	113	1,840*	Requesting publication of SIR after Examiner action	
115	110	215	55	Extension for response within first month	
116	400	216	200	Extension for response within second month	
117	950	217	475	Extension for response within third month	
118	1,510	218	755	Extension for response within fourth month	
128	2,060	228	1,030	Extension for response within fifth month	
119	310	219	155	Notice of Appeal	
120	310	220	155	Filing a brief in support of an appeal	
121	270	221	135	Request for oral hearing	
138	1,510	138	1,510	Petition to institute a public use proceeding	
140	110	240	55	Petition to revive unavoidably abandoned application	
141	1,320	241	660	Petition to revive unintentionally abandoned application	
142	1,320	242	660	Utility issue fee (or reissue)	
143	450	243	225	Design issue fee	
144	670	244	335	Plant issue fee	
122	130	122	130	Petitions to the Commissioner	
123	50	123	50	Petitions related to provisional applications	
126	240	126	240	Submission of Information Disclosure Stmt	
581	40	581	40	Recording each patent assignment per property (times number of properties)	
146	790	246	395	For filing a submission after final rejection (see 37 CFR 1.129(a))	
149	790	249	395	For each additional invention to be examined (see 37 CFR 1.129(a))	

Other fee (specify) _____ _____

Other fee (specify) _____ _____

SUBTOTAL (3) $ 0

*Reduced by Basic Filing Fee Paid

SUBMITTED BY: ORIGINAL SIGNED BY

Typed or Printed Name: Daniel M. De Vos

Signature _____ **Date** 6/21/98

Reg. Number 37,813 **Deposit Account User ID** _____ (complete if applicable)

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

In re Application of: Faan-Hoan Liu, et al. Serial No. To be assigned Filed: To be assigned For: A Method and Apparatus for Allowing Communication Between a Host Computer and at Least Two Storage Devices Over a Single Interface
This is a Continuation of: Serial No: 08/484,949 Filed: June 7, 1995
Which is a Divisional of: Patent No.: 5,446,877 Filed: June 23, 1994
Which is a Continuation of: Serial No.: 08/105,478 Filed: August 12, 1993
Which is a Continuation of: Serial No: 07/612,540 Filed: November 13, 1990

Examiner: To be assigned

Art Unit: To be assigned

Assistant Commissioner for Patents
Washington, D.C. 20231

PRELIMINARY AMENDMENT

Sir:

"Express Mail" mailing label number: EM088414194US
Date of Deposit: August 21, 1998
I hereby certify that I am causing this paper or fee to be deposited with the United States Postal Service "Express Mail Post Office to Addressee" service on the date indicated above and that this paper or fee has been addressed to the Assistant Commissioner for Patents, Washington, D. C. 20231
Conny Van Dalen
(Typed or printed name of person mailing paper or fee)
Conny VanDalen
(Signature of person mailing paper or fee)
8-21-98
(Date signed)

AMENDMENT

In the Claims:

Please cancel Claims 2-33, without prejudice.

In the Specification:

On page 1, before the first sentence, please insert – This is a continuation of application serial no. 08/484,949, filed June 7, 1995, which a divisional of U.S. patent no. 5,446,877, filed June 23, 1994, which is a continuation of application serial no. 08/105,478, filed August 12, 1993, which is a continuation of application serial no. 07/612,540--.

On page 5, line 9, please replace the word "tape" with --disk--.

On page 5, line 9, please replace the word "simultaneously" with --simultaneous--.

On page 10, line 25, please insert --111-- after the word "interface."

On page 12, line 21, please replace "107" with --108--.

On page 12, line 22, please replace "107" with --108--.

On page 12, line 25, please replace "107" with --108--.

On page 13, line 25, please replace "provides write to" with --provides a writeable--

.

On page 15, line 18, please replace the word "controlled" to --controller--.

On page 16, line 3, please insert the word --invention-- after the word "present."

On page 18, line 25, please replace the second occurrence of the word "tape" with the word -- disk--.

On page 25, line 10, please replace the word "write" with --writeable--.

On page 25, line 12, please replace "602 and count 603" with --603 and count 602--

.

On page 25, line 20, please insert the word --byte--after the words "formats a single address mark."

On page 26, lines 16-17, please replace "turn address mark on" with --"turn address mark on"--.

On page 26, line 19, please replace "turn address mark on" to --"turn address mark on"--.

REMARK

Entry of the above-listed amendment is respectfully requested. It is respectfully submitted that no new matter has been introduced by this preliminary amendment. Claim 1 remains in the application. Claims 2-33 have been canceled.

Invitation for a telephone interview

The Examiner is invited to call the undersigned at 408-720-8598 if there remains any issue with allowance of this case.

Charge our Deposit Account

Please charge any shortage to our Deposit Account No. 02-2666.

Respectfully submitted,

BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN LLP

Date: 9/21, 1998

Daniel M. De Vos
Reg. No. 37,813

12400 Wilshire Boulevard
Seventh Floor
Los Angeles, California 90025-1026
(408) 720-8598

58610.P003

UNITED STATES PATENT APPLICATION

FOR

TAPE BACKUP SYSTEM

Inventors:

Faan-Hoan Liu

Jorge Gustavson

Prepared by:

BLAKELY,SOKOLOFF,TAYLOR & ZAFMAN
12400 Wilshire Boulevard
Seventh Floor
Los Angeles, California 90025

(408) 720-8598

1. FIELD OF THE INVENTION

This present invention relates to data archival systems for use with computer systems and, more specifically, to a tape backup system compatible to be coupled with a hard disk controller on, for example, an IBM PC/AT or compatible.

2. BACKGROUND OF THE INVENTION

Virtually anyone who uses a computer system, and especially, a personal computer system, has considered the possibility of some hardware failure making data stored on that system inaccessible. The unfortunate among us have not only considered the possibility, but have had the possibility become a reality. Due to this possibility, it is today commonplace to employ some backup (or archival) system to periodically copy data stored in the computer system in order that the data may be restored at a later time in the event of such a hardware failure. In the personal computer world, most commonly, the backup system either employs so-called "floppy disks" onto which data is transferred, through the computers floppy disk drives, for storage or a tape drive which is coupled to the computer and which allows data to be communicated between the computer and tape drive for recording onto a tape medium (e.g., magnetic or optical tape).

In personal computers, such as the IBM PC and compatible family, known tape drives are coupled with the computer through a floppy disk interface. This design is employed largely to save the cost of requiring a separate controller for the tape drive and to avoid occupying one of the computer's slots with a separate tape controller. However utilizing the floppy controller as an interface presents certain problems in the operation and design of a backup system. For example, in the typical IBM PC or compatible, a floppy drive controller is capable of supporting up to two separate drives. Using one of the slots on the floppy drive controller to interface with a tape drive may be undesirable. Further, use of the floppy drive controller allows for a relatively slow interface data rate, e.g., 500 kilobits per second (Kb/s). Present day disk controllers may provide for relatively faster interface data rates, e.g., 5 or more megabits per second (Mb/s). Therefore, it

is desired to provide for coupling of a tape drive to a controller having a interface data rate more compatible with present day disk controllers.

Therefore, as one object of the present invention it is desired to provide a tape drive design which allows interfacing a tape drive to a computer system without need to interface through the floppy controller.

It is, of course, possible to provide for interfacing of a tape drive through a separate and relatively faster interface. However, in computer systems such as may utilize the preferred embodiment of the present invention, a separate interface requires use of one of a limited number of "card slots" on the motherboard of the computer system. Users of such computer systems generally prefer to avoid use of card slots whenever possible so that the card slots may be saved for other uses.

Therefore, as a second object of the present invention, it is desired to provide an interface for a tape backup system which does not require use of a separate card slot for interfacing with the host computer.

Recently, an interface has been developed known as the Intelligent Device Electronics (IDE) interface. The IDE interface typically allows for interfacing of at least two hard disk drives, each having an embedded controller. In typical operation, one disk drive may be addressed by the host computer through standard operating system routines in order to access data on that drive. During the time when the disk drive is executing the command to access data, the drive sets a bit indicating the interface is busy, preventing the host computer from issuing additional commands over the interface (e.g., a command to write data to the second device). Therefore, the IDE interface, utilizing standard operating system commands, is relatively incompatible with attempts to couple a tape drive to the device for purposes of "backing up" a disk drive coupled with the interface.

It is therefore, an object of the present invention to provide apparatus and methods for interfacing a tape drive with an IDE interface and to allow simultaneous operation of a disk drive and the tape drive coupled with the interface.

Finally, embedded controller circuits on present day disk drives designed for coupling with the IDE interface typically comprise standard, off-the-shelf, disk controller circuits, such as the Cirrus Logic CL-SH-260 disk controller circuit. It is desired to develop a tape drive utilizing standard, off-the-shelf, disk controller circuits. However, certain incompatibilities exist between outputs provided by such circuits and formats compatible for writing to industry standard tapes.

Therefore, as another object of the present invention, it is desired to provide circuitry for adapting outputs of standard, off-the-shelf, disk controller circuits to provide data compatible for writing industry standard tape formats.

SUMMARY OF THE DISCLOSURE

A data archival system is disclosed which is especially useful in computer systems, for example, in computer systems known as IBM PC's and compatibles therewith. The data archival system preferably comprises a tape backup device coupled with an Intelligent Device Electronics (IDE) interface such as may be available on an IBM PC/AT or compatible.

In particular, the present invention teaches a computer system having an interface for interfacing with at least a first storage device and a second storage device. Preferably the first storage device comprises a disk drive or similar random access storage apparatus. The second storage device preferably comprises a tape drive or similar data archival device. The first storage device comprises a first indicator having a first state for indicating the device is executing a command and a second state for indicating the device is not executing a command. The second storage device comprises a second indicator having a first state for indicating the second device is receiving information over the interface and a second state for indicating the second device is not receiving information over the interface.

Thus, the second device may receive information over the interface and during a period of time associated with receiving the information the second indicator may be in its first state thereby indicating the second device is receiving data. Upon completion of receiving the information, the second indicator may change states to indicate the second device is no longer receiving data. The second device may subsequently process the received information, such as by preferably accessing a storage medium such as tape and either reading or writing information thereto. At the same time, a command may be communicated to the first storage device over the interface; responsive to receiving the first command, the first storage device may set the first indicator means to a first state to indicate it is processing the command. For example, the command may instruct the device to read data from a disk storage media. Upon completion of the command, the first indicator means is returned to a state indicating the first storage device is not executing a command.

It is taught that the second device may continue executing a command after having completed receipt of information through the interface and after having set the second indicator to indicate completion of receipt of such information. For example, the second device may have received a command to write data to tape and, further, the second device may have received such data, for example, into a buffer memory. Subsequent to having received such a command and data, the second device may set its indicator to indicate it has completed receipt of the information and a processor may then direct a command to the first device to read data from a disk. The second device may then continue the process of writing data to tape simultaneously with the first device reading information from tape. Importantly, this simultaneously operation is accomplished while having both the first and second device coupled with a single interface.

The present invention further discloses use of a data archival application process for managing access to an interface having coupled thereto a disk drive and a tape drive wherein the application process initially replaces interrupt vectors pointing to standard ROM BIOS disk service routines with vectors pointing to routines in the data archival application. The application is then able to manage access to the disk service routines. Specifically, during operation the data archival process may call an operating system routine to, for example, read data from the disk for archival onto the tape. The operating system routine attempts to access the disk service routine through the appropriate intercept vector. However, as taught by the present invention, the application has replaced the address in the intercept vector with an address a routine in the archival application. Therefore, the data archival application intercepts the call to the disk service routine. The data archival application then switches control of the interface from the tape drive to the disk drive and then calls the ROM BIOS disk service routines.

In this manner, the data archival application may control simultaneous operation of the disk and tape drive while utilizing the ROM BIOS disk service routines to ensure compatibility.

In addition, a format adapter circuit is disclosed which advantageously allows use of industry standard disk controller circuits in a tape drive and provides for adapting output signals of such disk controller circuits to a format compatible with industry standard tape formats.

These and other aspects of the present invention will be better described with reference to the Detailed Description of the Preferred Embodiment and the accompanying Figures.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is a block diagram illustrating a computer system as may be utilized by the present invention.

Figure 2 is a block diagram illustrating coupling of a host computer through an IDE (Intelligent Drive Electronics) interface to a tape drive and a disk drive as is taught by the present invention.

Figure 3 is a block diagram illustrating components of a tape drive of the present invention.

Figure 4 is a block diagram further illustrating certain components of a tape drive of the present invention including a controller circuit, an IDE interface, a microprocessor, a RAM and a write/read interface.

Figure 5 is a diagram illustrating contents of a task file as may be implemented by the present invention.

Figure 6 is a diagram illustrating contents of a write control store as may be utilized by the present invention.

Figure 7(a) is an illustration of an encoder circuit and corresponding timing information as may be utilized by the present invention.

Figure 7(b) is an illustration of an decoder circuit and corresponding timing information as may be utilized by the present invention.

Figure 8 is a flow diagram illustrating an overall process for backing up a hard disk drive as may be utilized by the present invention.

Figure 9 is a flow diagram illustrating a backup loop of the backup process as may be utilized by the present invention.

Figure 10 is a flow diagram illustrating a method for reading information from disk as may be utilized by the present invention.

Figure 11 is a flow diagram illustrating a data restore process as may be utilized by the present invention.

Figure 12 is a flow diagram illustrating a restore data loop as may be utilized in the data restore process of the present invention.

Figure 13 is a flow diagram illustrating a method of writing information to a disk as may be utilized by the present invention.

Figure 14 is a diagram illustrating DOS address space as may be configured by the present invention.

Figure 15 is a flow diagram illustrating a prior art method of executing commands by a disk drive.

Figure 16 is a flow diagram illustrating a method of the present invention for executing commands by a tape drive.

DETAILED DESCRIPTION OF THE PREFERRED EMBODIMENT

A tape backup system is described. In the following description, numerous specific details are set forth such as specific circuits, etc., in order to provide a thorough understanding of the present invention. It will be obvious, however, to one skilled in the art that the present invention may be practiced without these specific details. In other instances, well-known circuits, structures and techniques have not been shown in detail in order not to unnecessarily obscure the present invention.

OVERVIEW

The computer system of the preferred embodiment

The overall computer system of preferred embodiment is described with reference to Figure 1. The preferred embodiment of the present invention is implemented on one of members of the IBM Personal Computer family or computer systems compatible with members of this family. In particular, the computer system of the present invention is implemented on a IBM PC/AT or compatible implementing the Intelligent Drive Electronics (IDE) interface 107 (also known as the "AT-Drive" interface or the ATA (AT attachment) interface. The IDE interface will be discussed in greater detail below. It will, of course, be obvious to one of ordinary skill in the art that any number of other computer systems may employ the teachings of the present invention.

In any event, a computer system as may be utilized by the preferred embodiment generally comprises a bus or other communication means 101 for communicating information, a processing means 102 coupled with the bus 101 for processing information, a random access memory (RAM) or other dynamic storage device 104 (commonly referred to as a main memory) coupled with said bus 101 for storing information and instructions for said processor 102, a read only memory (ROM) or other static storage device 106 coupled with said bus 101 for storing static information and instructions for said processor 102, an interface for allowing coupling of data storage devices, such as hard disk 107 (in addition, as will be seen, a data archival device such as a tape drive may

be coupled with the interface), with the bus 100 for storing information and instructions, a display device 122 , such as a cathode ray tube, liquid crystal display, etc, coupled to the bus 101 for displaying information to the computer user, an alphanumeric input device 125 including alphanumeric and other keys coupled to said bus 101 for communicating information and command selections to said processor 102, and a cursor control device 127, such as a mouse, track ball, cursor control keys, etc, coupled to said bus 101 for communicating information and command selections to said processor 102 and for controlling cursor movement. Finally, it is useful if the system includes a hardcopy device 129, such as a printer, for providing permanent copies of information. The hardcopy device 129 is coupled with the processor 102, main memory 104, static memory 106 and mass storage device 107 through bus 101.

The processor 102 of the preferred embodiment is one of the 80x86 microprocessor family manufactured by Intel Corporation of Santa Clara, California. In the preferred embodiment, the system is operated using a standard IBM PC-compatible operating system such as MS-DOS available from Microsoft Corporation of Beaverton, Oregon.

It will be apparent, from an understanding of the present invention from the below description, that several of the above-mentioned components of the computer system of the preferred embodiment are not essential to operation of a computer system employing aspects of the present invention. For example, as will be described in more detail below, the display 122, keyboard 125, cursor control device 127, and hard copy device 129 may not be present in certain implementations.

The IDE Interface

The IDE interface has evolved from earlier hard disk interfaces in which a hard disk adapter card, including a controller, was installed in a "slot" of a computer system. Such hard disk adapter cards are typically capable of supporting up to two hard disk drives, although only one drive may be written to or read from at a time. The two drives interface to the card through an interface

known as the ST506 interface. More recently developed hard disk drives comprise an embedded controller and processor removing the requirement for the adapter card to include its own controller. The interface allowing for use of embedded controllers in a hard disk is known as the IDE or Intelligent Device Electronics interface. The programming model and command codes for the IDE interface are the same as for the ST506 interface; however, as controllers are embedded in the drives themselves, the IDE interface must allow for two controllers to be coupled with the interface and must provide for proper management of information transferred. To manage access to the drives, the IDE interface depends on a status bit (the BUSY bit) in a set of registers (the "task file") associated with each drive. The task file 500 and the BUSY bit 521 of the preferred embodiment are described in greater detail below with reference to Figure 5.

Figure 2 illustrates in somewhat greater detail the coupling of a data archival device 108 and a hard disk storage device 107 with an IDE interface 111 of a host computer 201. In particular, the data archival device 108 and hard disk storage device 107 are both coupled to communicate data with the host computer 201 over data lines 202. The preferred embodiment provides 16 lines for communication of data between the host computer 201 and devices 107 and 108. In addition, devices 107 and 108 are coupled to receive 3 bits of address information on lines 203 and to receive control information on line 204 through interface 111. The control information comprises read/write strobes, chip select control information and ready signals.

Finally, the devices 107 and 108 are coupled to provide interrupts to the host computer through interface 111 on line 205.

The circuitry of the data archival device 107 of the preferred embodiment is described in greater detail with reference to Figure 3. Figure 3 illustrates the device 107 comprises a connector 301 for coupling with the IDE interface 111 of the host computer 201. The connector 301 is preferably a 40-pin connector and comprises 16 data pins for communicating data information between a controller circuit 302 of the device 107 and the interface 111 of the host computer 201; 3 address pins for receiving address information from the host computer 201 for communication to

the circuit 302; a control pin for receiving control information from the computer 201 for communication to the circuit 302; an interrupt pin for signalling interrupts generated by the circuit 302 to the computer 201; and a ready signal line for communicating ready status to the computer 201.

The controller circuit 302 is responsible for communicating information with the host computer over connector 301; for managing access to RAM buffer 303; and for formatting data to be written by the device. In the preferred embodiment, part number CL-SH260 Hard Disk Controller available from Cirrus Logic, Inc. of Milpitas, California is utilized. However, in alternative embodiments it is recognized that alternative circuitry may be utilized. For example, functions of the CL-SH260 may be implemented in discrete circuitry, in a gate array or in a programmed logic array. Further, alternative commercially available circuits may be employed which provide similar or compatible circuity. For example, part number AIC-6060 hard disk controller available from Adaptec may be substituted as a compatible part.

The controller circuit 302 is illustrated in greater detail with reference to Figure 4. As previously stated, the controller circuit 302 provides the functions of interfacing with the host computer through host interface section 402; of providing for buffer management of RAM buffer memory 303 through buffer manager section 404; and of providing for formatting data through formatter section 403. The particular workings of each of these sections will be well understood by one of ordinary skill in the art, especially with an understanding of the CL-SH260 controller circuit as utilized by the present invention. Further information on the CL-SH260 controller is available from Cirrus Logic, Inc. as CL-SH-260 Integrated PC XT/AT Disk Controller Data Sheet and Application Notes AN-SH1 & AN-SH5. However, it is worthwhile to discuss here that the controller 302 provides registers for the task file 500 as part of host interface 500. The task file register 500 will be described in greater detail below with reference to Figure 5. Additionally, formatter section 403 provides write control store circuitry 409 for storage of control information. Write control store 409 will be discussed in greater detail below with reference to Figure 6.

The controller circuit 302 is coupled to communicate control, data and interrupt information with processor 304 and gate array 307. Processor 304 and gate array 307 are provided to control the drive electro-mechanics such as circuitry for controlling tape movement. In the preferred embodiment, the processor 304 is an Intel 8052 microprocessor. Again, it will be apparent to one of ordinary skill in the art that alternative embodiment may employ alternative processing means to provide for control of the controller circuit 302.

In addition to providing for control of the drive electro-mechanics, processor 304 is programmed, via 32K ROM 305, to provide certain control over controller circuit 302. Of particular interest to the present invention, and as will be discussed in greater detail below, processor 304 is programmed to set the BUSY bit whenever information (i.e., a command or data) is being received over the interface and to clear the BUSY bit after the information has been received.

The formatter section 403 of controller 302 is further coupled to provide output information to encoder 311 and to receive input information from decoder 312. In the preferred embodiment, controller 302 is designed to provide output information to encoder 311 and to receive input information from decoder 312 in a format commonly referred to as modified FM or MFM. Encoder 311 is provided to convert the MFM information to non-return to zero (NRZ) format in a well known fashion. Decoder 312 is provided to convert NRZ information back to MFM information for presentation to the controller 302, again in a well known fashion. Encoder 311 and decoder 312 are implemented as programmed logic arrays in the system of the preferred embodiment. Although it is known in prior art hard disk drive systems to provide circuitry for encoding MFM signals and decoding NRZ signals, certain advantages of the encoder 311 and decoder 312 allowing for use of the circuitry of the present invention with a data archival system and, specifically, with a tape drive format as utilized by the preferred embodiment. These advantages will be described in greater detail below with reference to Figures 6, 7(a) and 7(b).

The Task File

As can be seen with reference to Figure 5, the task file 500 of the preferred embodiment of the present invention comprises ten separate eight-bit registers which may be referred to as (1) the status register 501; (2) the alternate status register 502; (3) the drive select register 503; (4) the error code register 504; (5) the segment number (low) register 505; (6) the segment number (high) register 506; (7) the segment count register 507; (8) the mode register 508; (9) the data register 509; and (10) the digital input register 510.

The Status Register 501

The status register comprises eight indicator bits. Of particular interest to the present invention is the BUSY bit 521 (bit 7 counting from bit 0 at the right). In a typical disk drive operation in the prior art, at the beginning of a command sequence, the BUSY bit 521 is set under the control of a local processor. The BUSY bit 521 remains set until completion of an operation. During the time period when the BUSY bit 521 is set, the host computer is not allowed access to the remaining registers in the task file 500. Further, the host computer may not access any other device coupled with the IDE interface during the time period when one controller coupled with the bus has its BUSY bit 521 set. In essence, the BUSY bit 521 is used as a collision avoidance mechanism and it is assumed in prior art systems utilizing the IDE interface that it is not desirable, and in fact, it is not allowed, to access two devices coupled with the IDE interface simultaneously. After the BUSY bit 521 is cleared, the controlled is not allowed to change the contents of the remaining registers of the task file 500. Thus, it is understood that, in prior art systems, the use of the BUSY bit 521 as a contention avoidance mechanism prevents access to one device coupled with an IDE interface when a second device, also coupled with the same IDE interface is busy executing a command. In essence, the interface simulates a situation in which a single controller is available to control access to both devices.

It should be noted that the IDE interface allows for one special case, the SEEK command, in which the controller clears the BUSY bit 521 after beginning execution of the command and prior to its completion. Therefore, a second drive coupled with the IDE interface may be accessed

while the SEEK command is executing. Implementation of this special case in the IDE interface requires use of a special status bit (SEEK Complete) in the task file of the disk drive.

As will be seen, the present provides capability for simultaneous reading and writing access to a data archival device and a hard disk, both attached to a single IDE interface of a host computer.

The status register is also used as a command register and is written with the command to be executed.

The Drive Select Register 503

The drive select register 503 is written by the host computer to select either one of the two drives coupled with the IDE interface before programming of the task file registers and issuing commands. In the preferred embodiment, a hard disk drive may be attached as drive 0 (also referred to as the "master") and a tape drive may be attached as drive 1 (also referred to as the "slave"). In certain embodiments, such as a system with a secondary IDE channel, the tape drive could be coupled as drive 0. Use of the BUSY bit will be discussed in greater detail below.

In the preferred embodiment, a drive select bit 522 (bit 4 counting from bit 0 being the far right hand bit) is set or cleared to select the desired drive. The remaining bits (bits 0-3 and 5-7) are not used in current implementations.

The Alternate Status Register

The alternate status register 502 is identical to the status register 501; however reading the alternate status register does not acknowledge and clear interrupt requests which are acknowledged and cleared by reading the status register 501. This register is useful where it is desired to be able to inquire into the device status without losing a pending interrupt.

The Error Code Register

The error code register 504 is a two field register indicating error information: an error report pending status bit and a 7-bit error code.

The Data Register

The data register 509 is utilized to allow direct interface with the tape controller's buffer management logic to allow reads or writes with the drives buffer RAM memory.

The Segment Count Register

The segment count register 507 is loaded with a segment count prior to issuing a read or a write command. If a count of zero is loaded, it is interpreted as an indefinite count and the command will continue until terminated by another command. During the transfer of data, the segment count is updated by the tape drive after each segment is transferred, unless an indefinite count is specified.

The Segment Address Registers

The segment address registers 505 and 506 are loaded with a beginning segment address prior to reading or writing to the tape. The segment address is a 16-bit value with the low order eight bits stored in register 505 and the high order eight bits stored in register 506. During multiple segment read and write operations, the segment address is incremented as each transfer is completed.

The Mode Register

The mode register 508 comprises a set of bits to control and report operational modes of the tape drive of the preferred embodiment including a format code, length of tape code, tape cartridge present code, tape cartridge write protected code, interrupt mode select, and a data packing method code.

SIMULTANEOUS OPERATION OF THE TAPE AND DISK DRIVES

Prior art methods of operating data archival devices in, for example, an IBM PC or compatible, provide for coupling the tape drive through a interface separate from the interface for the hard disk. For example, it is common to couple a tape drive to a floppy disk interface. In this way, one interface is available for allowing access to the disk drive while a separate interface is available for allowing access to the tape drive. In this arrangement, the respective interfaces may operate to allow simultaneous operation of the tape and disk drives. However, in certain computer

systems it is desirable for a number of reasons to allow for coupling of both a tape and disk drive through a single interface.

Therefore, as one aspect of the present invention it is disclosed to allow coupling of both a tape drive and a disk drive to a single interface while providing for simultaneous operation of both the tape and disk drive.

The BUSY Bit

Known devices (e.g., disk devices) which interface with the IDE interface operate by setting their BUSY bit in their status register when the device is executing a command. A well-known processing utilized by disk drives which interface with the IDE interface is illustrated with reference to Figure 15. First, the host processor sets the drive select bit 522 to select the disk (e.g., in a typical configuration, the host processor sets the drive select bit to 0), block 1522. The host proceeds to program the task file registers of the disk drive to execute the command, block 1502. After the task file registers are programmed by the host, an on-board processor on the drive sets the BUSY bit 521 and begins executing the command. Execution of the command may include communicating (e.g., reading or writing) information over the interface. For example, data to be written to the disk may be communicated by the host processor over the interface to the disk drive. The disk drive may initially accept the data into a buffer memory and then retrieve the data from the buffer memory for writing to disk. Typically, the buffer memory is a higher speed device than the disk itself and, therefore, the host computer may complete transmission of the information before the drive completes execution of the command to write the data to disk. In any event, eventually the disk completes execution of the command, block 1505. Upon completion of the command, the on-board processor clears the BUSY bit, block 1506.

During the period the BUSY bit was set (i.e., from block 1503 through block 1506), the host processor is prevented from accessing registers 503-509 of the task file registers on both the disk drive and tape drive. As a result, during the period in time when the tape drive is executing the

command, even after it may have received all necessary information over the interface, the host computer is prevent from accessing the tape drive.

Figure 16 illustrates a method of the present invention for processing commands to the tape drive. Similar to the discussion in connection with Figure 15, the host processor initially sets the drive select bit to select the tape drive, block 1601, and the host processor programs the tape drives task file register, block 1602. The tape drives on-board processor then sets the BUSY bit 1603 and begins executing the command. For example, the command may be a request to write information to tape, in which case, information is communicated by the host processor, over the interface, to the tape drive, block 1604. After completion of communicating information over the interface, the tape drives on-board processor clears the BUSY bit, block 1605. In the exemplary case of a request to write information to tape, the tape drive may accept information to be written over the interface and temporarily store the information in a relatively high speed buffer memory prior to the information being written to tape. Therefore, the tape drive may not have completed processing of the command prior to clearing the BUSY bit. In any event, at some point in time the tape drive completes execution of the command, block 1606; however, as has been explained, the BUSY bit has been cleared prior to completion of execution of the command. In this way, the host may begin communications with the disk drive while the tape drive completes execution of the command "off-line".

The process of backing up and restoring information in the system of the preferred embodiment will be explained in greater detail with reference to Figures 8-14.

The Backup Process

Figure 8 is an overall flow diagram of a backup (or data archival) process as may be utilized by the present invention. In the preferred embodiment, an application program is provided to manage the backup process. The application program is initialized in any of a number of well-known manners, block 801. For example, a user may input a command into the host computer to initialize or start-up the application process.

When the application program initializes it determines and stores the addresses of interrupt vectors for the system hard disk interrupt (referred to as INT 14 in the MS-DOS operating system) and the ROM BIOS hard disk service routine (referred to as INT 13 in the MS-DOS operating system). The application program then installs the address of certain of its own routines in place of the addresses of the interrupt vectors; effectively the application program takes control of the disk service routines, block 802. This may be better understood with reference Figure 14.

Figure 14 illustrates certain portions of the address space in an MS-DOS system. The address space includes certain interrupt vectors 1401 including a hard disk interrupt vector (hardware interrupt vector 1402) and a ROM BIOS hard disk service vector (INT 13H vector 1403). During typical operations, vector 1402 is an address pointer to the ROM BIOS disk interrupt handler 1423 in the ROM BIOS code 1421 and vector 1403 is a pointer to ROM BIOS INT 13 disk services handler 1422, again in the ROM BIOS code. However, as mentioned above, when the tape application program initializes, interrupt vector 1402 is changed to point to an interrupt handler routine 1413 in the tape application program 1411 and interrupt vector 1403 is changed to point to a "gatekeeper" routine 1412 in the application program 1413.

As will be explained in greater detail below, these routines allow the application to have its own driver routines for interfacing with the tape drive and to use the ROM BIOS routines for accessing the disk.

The "gatekeeper" routine 1412 ensures that when the operating system (MS-DOS) is accessing the disk, all hardware interrupts are directed to the ROM BIOS's routines for servicing and when the disk is not being accessed, the IDE bus is available for tape drive commands and interrupts.

The IDE interrupt gatekeeper and tape drive interrupt function 1413 allows the application to redirect the hardware interrupt vector 1402 to point to the disk service routine when disk drive operations are being performed and to use the applications own interrupt routines when the tape operations are being performed.

Referring back to Figure 8, the application program then issues a command to the tape drive to write data to tape (i.e., to backup data) by programming its task file registers, block 803. The application then directs archiving of data from the disk drive to the tape drive by reading data from the disk, blocking the data into 32K blocks, and transferring the data to tape. These functions will be explained in greater detail with reference to Figures 9 and 10.

Upon completion of the archival process, the application releases control of the disk service routines by writing the stored addresses for vectors handlers 1422 and 1423 back to vectors 1402 and 1403, respectively, block 805. The application then terminates, block 806.

Referring now to Figure 9, the processing of backing up data from the disk is essentially a loop in which, first, data is read from the disk, block 901; assuming there is more data to read, block 902, the data is blocked into 32K blocks and transferred to the tape drive, block 904. The process of reading data from the disk is explained in greater detail with reference to Figure 10. This process essentially entails the "gatekeeper" process referred to earlier. Initially, the application program uses operating system calls, in the conventional manner, to request data be read from the disk. The operating system calls attempt to call the disk service routine (INT 13) at the address given by vector 1403. As discussed earlier, the application program had previously replaced the address of INT 13 with the address of gatekeeper function 1413. Therefore, the call to the disk service routine is intercepted by the application program, block 1003.

The application program then switches access to the IDE bus from the tape drive to the disk drive (the state of the drive select bit in the drive select register of both the tape and disk drive is inverted, e.g., changed from a 1 to a 0). The tape drive is then disabled from accepting further data transmitted on the bus by clearing the data request status bit 523. In this way, the tape drive is controlled to prevent it from mistakenly accepting data transferred on the interface from the disk drive to the host computer.

As previously noted, the application program has stored the address of the ROM BIOS INT 13 disk services routine. At this point, having disabled the tape from receiving further data and

having set the disk select bit 521 to allow the disk drives task file registers to be programmed, the application program calls the ROM BIOS INT 13 disk services routine to initiate transfer of data from the disk to the host. The ROM BIOS INT 13 routine programs the disks task file registers and the disk begins execution of the command. As discussed previously with reference to Figure 15, the disk will set its BUSY bit prevent other accesses to the bus during the time it is executing the command. Upon completion of execution of the command, the disk clears the BUSY bit and the ROM BIOS INT 13 routine returns control to the operating system routine. The operating system routine then returns control to the application process, block 1006.

The application program then proceeds to switch communications back to the tape drive by switching the state of the disk select bit (e.g., from a 0 to a 1), block 1007. The tape drives DMA transfer is then again enabled, block 1008, and the application process continues to transfer data to the tape.

The Restore Process

Figures 11-13 illustrate the process of the preferred embodiment for restoring data from the tape to disk. The restore process is similar to the backup process and begins by initializing the application program, block 1101 and by the application process taking control of the disk services routine, block 1102. The application program then programs the tape drives task file registers to read data from tape, block 1103 and the tape drive begins reading data from the tape and transferring the data to its buffer memory 303 under control of controller 302. The tape drive then transfers the data across interface 301 to the host computer and the application program transfers the data to disk, block 1104 and continues this process until all requested data has been restored (see Figure 12). Upon completion of the restore process, the application process releases control of the disk services processes, block 1105, and terminates, block 1106.

The process of transferring data from tape to disk of block 1104 is illustrated in greater detail with reference to Figure 12 which illustrates that initially data is read from tape 1201. In the preferred embodiment, data is read from tape using the method and mechanisms described above.

That is, first a command is issued by the host processor to the tape drive. The tape drive then proceeds to read data from tape and to store the data in its internal buffer. This process may continue without requiring access to the IDE interface (i.e., the tape drive is "offline"). As data is placed in the buffer it may be transferred to the host computer main memory 104 using a DMA transfer facility, block 1203. During the period of time that data is actually being transferred, the tape drive indicates to the host computer that the IDE interface is unavailable for other uses by setting its BUSY bit. After the data is transferred from the data buffer 303 of the tape drive to the main memory 104 of the host computer, the tape drive clears the BUSY bit allowing the tape application software to again control access to the bus.

The tape application software then begins the process of writing the data from the main memory 104 to disk, block 1204. During the period of time that data is being written to disk, the tape drive continues to process the read tape command and to store data in its on-board buffer memory 303.

Figure 13 illustrates the process of writing data to disk, block 204, in greater detail. This process is similar to the process of Figure 10 which illustrated reading data from disk. Initially, the tape application software calls standard operating system routines for accessing the tape drive. The operating system routines attempt to call the disk service routine, block 1301. As has been explained, especially with reference to Figure 8, 11 and 14, the application program replaced the interrupt vectors of the ROM BIOS disk service routines with vectors to routines in the application process. Therefore, the calls by the operating system routines to the disk service routines are intercepted by the application process, block 1302. The application process then switches access to the IDE bus to disk, block 1303 by setting the disk select bit 522.

To the extent not already specifically discussed, it should be noted that the disk select bit 522 provides for control of which device coupled with the IDE bus is to be accessed. When the host writes the drive select register 503 to alter the state of the drive select bit, the task file registers of both the disk and tape drive are effected. Likewise, when other registers are written by the host,

the task files of both drives are effected irregardless of which drive is selected by the drive select bit 522. In the system of the preferred embodiment, when issuing a command, the host computer first determines that the IDE bus is available by determining the state of the BUSY bit, 521. The host then writes the disk select register, 522 followed by the other registers needed for the operation and finally writes the command register 501 (as stated previously, the command register is a dual purpose register and is used as the status register while the drive is executing a command). The drives are each programmed such that only the drive selected by the drive select bit 522 will respond to the command. As soon as the command is written, the BUSY bit is set and the operation proceeds as has been described.

Again referring to Figure 13, the tape drive responds to being deselected by disabling DMA transfers, block 1304, thus preventing inadvertent writing of data into its buffer memory during transfers on the IDE bus directed to the disk.

The application software then calls the ROM BIOS INT 13 disk service routine, block 1305. As a reminder, the application process has previously saved the address of this routine and, thus, has the address available to call directly without having to access the interrupt vector 1403. The disk service routine handles writing of data to the disk drive over the IDE interface. During the time that the write data command is being processed, the disk drive sets the BUSY bit. During the period of time when the BUSY bit is set, the disk drive performs DMA transfers from main memory 106.

Upon completion of writing of data, the disk service routine returns control to the application program, block 1306. The application program then writes the disk select bit 522 to select the tape drive, block 1307, and the tape drive again enables DMA writes from the tape's buffer memory to the main memory 104.

Thus, it can be seen that the present invention allows for effective simultaneous access to two independently controlled devices both coupled with a host computer through a single interface. This simultaneous access provides for improved performance of the devices, especially as

described in connection with the system of the preferred embodiment, during tape backup and restore processes.

TAPE DRIVE INTERFACE WITH STANDARD DISK CONTROLLER CIRCUITS

As one aspect of the present invention, it is taught to utilize industry standard disk drive controller circuits for control and interfacing of a tape drive. For example, as has been previously discussed, in the preferred embodiment a Cirrus Logic CL-SH-260 disk controller circuit is utilized as controller circuit 302. Responsive to the host interface 402 portion of the controller circuit 302 receiving a command from the host computer, the formatter portion of the circuit writes instructions to the write control store 409. An exemplary set of instructions is given with reference to Figure 6. The instructions may include such information as a command 601, a segment address 602 and count 603. Initially, a command may be given such as command 0001 to begin tape movement, followed by a command such as a synchronize command 0002, a write address mark command (WAM), and a command to begin writing data from the buffer memory 0004.

It is noted that in the data archival system of the present invention it is highly desirable to provide for compatibility with existing tape formats. Unfortunately, such existing tape formats are not compatible with standard disk drive formats, and especially with the write formats produced by standard controller circuits such as the Cirrus Logic CL-SH-260.

One area of incompatibility is in the the area of writing address marks. In writing address marks to standard disk drive formats a single address mark is typically written. The standard controller circuits begin CRC calculations when the write address mark command is issued. Standard tape drive formats require writing of three consecutive address marks to tape. Unfortunately, in order to provide for accurate calculation of the CRC, it is not desirable have the controller circuit 302 issue three separate write address mark instructions. Therefore, what is desired is to provide circuitry which will effectively provide for separation of the write address mark and "start CRC" instructions in a standard disk controller circuit such as the SH-260.

The present invention implements such circuitry by providing in the NRZ to MFM encoder circuit 311 circuitry to effectively write three address marks each time an address mark is received from the controller circuit 311. This circuitry may be referred to as an address mark generator. In the preferred embodiment, encoder 311 is implemented as in a programmed logic array. In a similar fashion, when reading information from tape, the controller circuit is programmed to start CRC calculations upon receipt of each address mark. However, when reading standard tape formats, as has been discussed, three consecutive address marks are read from tape and it is required to begin calculation of the CRC with the first address mark and not to recalculate the CRC for each of the address marks. Therefore, the MFM to NRZ decoder 312 of the present invention provides address mark detection circuitry for detecting address marks and removing from the information transmitted to the controller two of three address marks in a sequence of three consecutive address marks. This is better illustrated with reference to Figure 7(a) and Figure 7(b).

Figure 7(a) illustrates the encoder 311 which receives NRZ (non-return to zero) information from the controller circuit 302 and produces MRM (modified FM) output. The encoder includes circuitry 701 (coded in a PLA circuit) for detecting instructions to turn on the address mark on the NRZ input. In the prior art, as illustrated by signal 711, each turn address mark on instruction will result in one address mark being generated on the MFM output to the drive. The present invention provides for generation of three address marks responsive to each turn address mark on instruction as illustrated by signal 712.

In a similar fashion, as illustrated by Figure 7(b), the present invention provides address mark detection circuitry which detects sequences of three address marks being read from the drive and being supplied on the MFM input to the decoder 312, such as the three address marks illustrated by signal 732. Responsive to detecting three consecutive address marks, decoder 312 provides a single address mark output on the NRZ output to controller circuit 302.

In this way, the present invention provides for compatibility with standard disk drive controller circuits allowing interface with the IDE bus while providing for compatibility with the format of existing tape backup systems.

Thus, what has been described is an improved apparatus and method for providing for data archival in a computer system. The described method is especially useful in computer systems utilizing the Intelligent Device Electronics (IDE) interface, such as may be utilized by IBM PC-ATs or compatibles.

CLAIMS

What is Claimed is:

1. A computer system comprising:

   (a) an interface for interfacing with at least a first storage device and a second storage device;

   (b) said first storage device coupled with said interface;

   (c) first indicator means coupled with said first storage device, said first indicator means having a first state for indicating said first storage device is executing a command and having a second state for indicating said first storage device is not executing a command;

   (d) said second storage device further coupled with said interface;

   (e) second indicator means coupled with said second storage device having a first state for indicating said second storage device is receiving information over said interface and having a second state for indicating said second storage device is not receiving information over said interface.

2. The computer system as recited by claim 1 wherein said first storage device is a hard disk device.

3. The computer system as recited by claim 2 wherein said second storage device is a tape drive device.

4. The computer system as recited by claim 1 wherein said interface is an Intelligent Drive Electronics (IDE) interface.

5. The computer system as recited by claim 4 wherein said first indicator means comprises a bit in a task file register coupled with said first storage device.

6. The computer system as recited by claim 5 wherein said second indicator comprises a bit in a task file register coupled with said second storage device.

7. A computer system comprising:

(a) an interface having a first slot for coupling with a first device and a second slot for coupling with a second device;

(b) a hard disk drive coupled with said interface through said first slot;

(c) a tape drive coupled with said interface through said second slot, said tape drive comprising a first circuit receiving information from said interface and formatting said information for storage on a medium, said first circuit including an indicator having a first state for indicating said tape drive is receiving information over said interface and a second state for indicating said tape drive is not receiving information over said interface.

8. The computer system as recited by claim 7 wherein said interface is an Intelligent Drive Electronics (IDE) interface.

9. The computer system as recited by claim 8 wherein said indicator is a bit in a task file register.

10. The computer system as recited by claim 9 wherein said first circuit further comprises a buffer management means and said tape drive further comprises a buffer memory managed under the control of said buffer management means.

11. The computer system as recited by claim 10 wherein said tape drive further comprises processor means for controlling said first circuit.

12. A computer system comprising:

(a) an 8086 family processor;

(b) a random access memory coupled with said 8086 family processor;

(c) a device interface having a first slot and a second slot;

(d) a central bus for communication information between said 8086 family processor, said random access memory and said device interface;

(e) a hard disk drive coupled with said first slot;

(f) a tape drive coupled with said second slot.

13. The computer as recited by claim 12 wherein said tape drive comprises an indicator having a first state for indicating said tape drive is receiving information over said interface and a second state for indicating said tape drive is not receiving information over said interface.

14. The computer system as recited by claim 13 wherein said device interface is an Intelligent Device Electronics interface.

15. The computer system as recited by claim 14 wherein said tape drive comprises a Cirrus Logic CL-SH-260 controller circuit.

16. In a computer system comprising a host computer having an interface for coupling with a first data storage device and a second data storage device, a method of operating a data archival apparatus comprising the steps of:

(a) said host computer setting a first indicator on said first and second devices, said first indicator indicating said host computer will communicate with said first device;

(b) said host computer communicating first command information to said first device;

(c) setting a second indicator to indicate said interface is busy, said indication preventing said host from communicating command information over said interface;

(d) said host computer communicating information for storage on said first device over said interface;

(e) said first device receiving said information and storing said information in a buffer memory;

(f) completing communication of said information;

(g) clearing said second indicator thereby allowing said host computer to communicate further commands over said interface;

(h) said first device continuing to execute said first command by reading information from said buffer memory and storing said information on a media.

17. The method as recited by Claim 16 wherein said first device is a tape drive and said second device is a disk drive.

18. The method as recited by Claim 17 wherein said host computer is an IBM PC or compatible computer.

19. The method as recited by Claim 18 wherein said interface is the Intelligent Device Electronics interface.

20. In a computer system having a host computer and an interface for coupling a first device and a second device, said host computer including an address space having a interrupt vector address space, a input/output system address space, and an application address space, said interrupt vector address space including a first pointer to a first disk service routine in said input/output address space, said first pointer addressable at a first address, a method of operating a data archival apparatus comprising the steps of:

(a) said host computer initializing a data archival process;

(b) said data archival process reading said first pointer at said first address and storing the value of said first pointer;

(c) said data archival process storing a second pointer at said first address, said second pointer pointing to a second disk service routine in said data archival process;

(d) said data archival process archiving data from said second device to said first device by repetitively issuing commands to read data from said second device and to write data to said first device;

(e) said data archival process writing said stored value of said first pointer to said first address upon completion of said step d; and

(f) terminating said data archival process.

21. The method as recited by Claim 20 wherein said step of reading data from said second device comprises the steps of:

(g) said data archival process calling an operating system routine to read data from said second device;

(h) said operating system routine attempting to call said first disk service routine by accessing said first pointer;

(i) said second disk service routine being called responsive to said step h;

(j) said second disk service routine providing access to said interface to said second device;

(k) said second disk service routine calling said first disk service routine, said first disk service routine managing access to said second device to read data from said second device;

(l) completing said step of reading data from said second device;

(m) said first disk service routine returning control to said first disk service routine;

(n) said first disk service routine providing access to said interface to said first device.

22. The method as recited by Claim 21 wherein said first device is disabled from performing DMA data transfers while reading data from said second device.

23. In a computer system having a host computer and an interface for coupling a first device and a second device, said host computer including an address space having a interrupt vector address space, a input/output system address space, and an application address space, said interrupt vector address space including a first pointer to a first disk service routine in said input/output address space, said first pointer addressable at a first address, a method of operating a data archival apparatus comprising the steps of:

(a) said host computer initializing a data restore process;

(b) said data restore process reading said first pointer at said first address and storing the value of said first pointer;

(c) said data restore process storing a second pointer at said first address, said second pointer pointing to a second disk service routine in said data restore process;

(d) said data restore process restoring data from said first device to said second device by repetitively issuing commands to read data from said first device and to write data to said second device;

(e) said data restore process writing said stored value of said first pointer to said first address upon completion of said step d; and

(f) terminating said data restore process.

24. The method as recited by Claim 23 wherein said step of writing data to said second device comprises the steps of:

(g) said data restore process calling an operating system routine to read data from said second device;

(h) said operating system routine attempting to call said first disk service routine by accessing said first pointer;

(i) said second disk service routine being called responsive to said step h;

(j) said second disk service routine providing access to said interface to said second device;

(k) said second disk service routine calling said first disk service routine, said first disk service routine managing access to said second device to write data to said second device;

(l) completing said step of writing data to said second device;

(m) said first disk service routine returning control to said first disk service routine;

(n) said first disk service routine providing access to said interface to said first device.

25. The method as recited by Claim 24 wherein said first device is disabled from performing DMA data transfers while writing data to said second device.

26. A data archival process for use in a computer system, said computer system having a host computer having an Intelligent Device Electronics (IDE) interface for coupling a first device and a second device, said host computer including an address space having a interrupt vector address space, a input/output system address space, and an application address space, said interrupt vector address space including a first pointer to a first disk service routine in said input/output address space, said first pointer addressable at a first address, said computer

system further including a tape drive coupled with said computer system as said first device and a disk drive coupled with said IDE interface as said second device, a method of operating a data archival apparatus comprising the steps of:

(a) said host computer initializing a data archival process;

(b) said data archival process reading said first pointer at said first address and storing the value of said first pointer;

(c) said data archival process storing a second pointer at said first address, said second pointer pointing to a second disk service routine;

(d) said data archival process archiving data from said disk drive to said tape drive by repetitively issuing commands to read data from said disk drive and to write data to said tape drive;

(e) said data archival process writing said stored value of said first pointer to said first address upon completion of said step d; and

(f) terminating said data archival process.

27. The method as recited by Claim 26 wherein said step of reading data from said disk drive comprises the steps of:

(g) said data archival process calling an operating system routine to read data from said disk drive;

(h) said operating system routine attempting to call said first disk service routine by accessing said first pointer;

(i) said second disk service routine being called responsive to said step h;

(j) said second disk service routine providing access to said interface to said second device;

(k) said second disk service routine calling said first disk service routine, said first disk service routine managing access to said disk drive to read data from said disk drive;

(l) completing said step of reading data from said disk drive;

(m) said first disk service routine returning control to said first disk service routine;

(n) said first disk service routine providing access to said interface to said tape drive.

28. The method as recited by Claim 27 wherein said tape drive is disabled from performing DMA data transfers while reading data from said disk drive.

29. A device for communicating information to a media comprising:

a formatter circuit for receiving information from a host computer in a first format and converting said information to a second format, said second format compatible for writing to a hard disk device;

a format adapter circuit for receiving information in said second format and adapting said second format to a third format compatible for writing to a tape drive.

30. The device as recited by Claim 29 wherein said formatter circuit comprises a Cirrus Logic SH-260 controller circuit.

31. The device as recited by Claim 29 wherein said formatter circuit provides in said second format a signal for turning on writing of an address mark on said media and said format adapter circuit comprises circuitry for receiving said signal for turning on writing of said address mark and provides in said third format three consecutive address marks for writing to said media.

32. A tape backup device compatible for coupling with an IDE interface, said tape backup device comprising a circuit for receiving input signals in a format compatible for writing to a tape drive and for providing output signals in a format compatible for writing data to a tape.

33. The tape backup device of Claim 33 wherein said input signals include a write address mark signal, said circuit receiving said write address mark signal and providing as said output signal three write address mark signals.

ABSTRACT OF THE DISCLOSURE

A data archival device having an embedded controller compatible with a host computer interface for disk drives having embedded controllers. The data archival device provides for simultaneous writing of data to a tape media coupled with an interface of a host computer while reading of data from a disk coupled with the interface. This simultaneous access to both the tape and disk drives is accomplished through use of an application process for managing access to the tape drive, as well as controlling the tape drive to "own" the interface only during periods of time when information is being communicated over the interface. The tape drive may receive information over the interface into an on-board buffer and then release the interface to allow the host computer to access the disk drive. The tape drive may then continue "off-line" to write data to a tape media or to perform other functions not requiring communication of information over the interface.

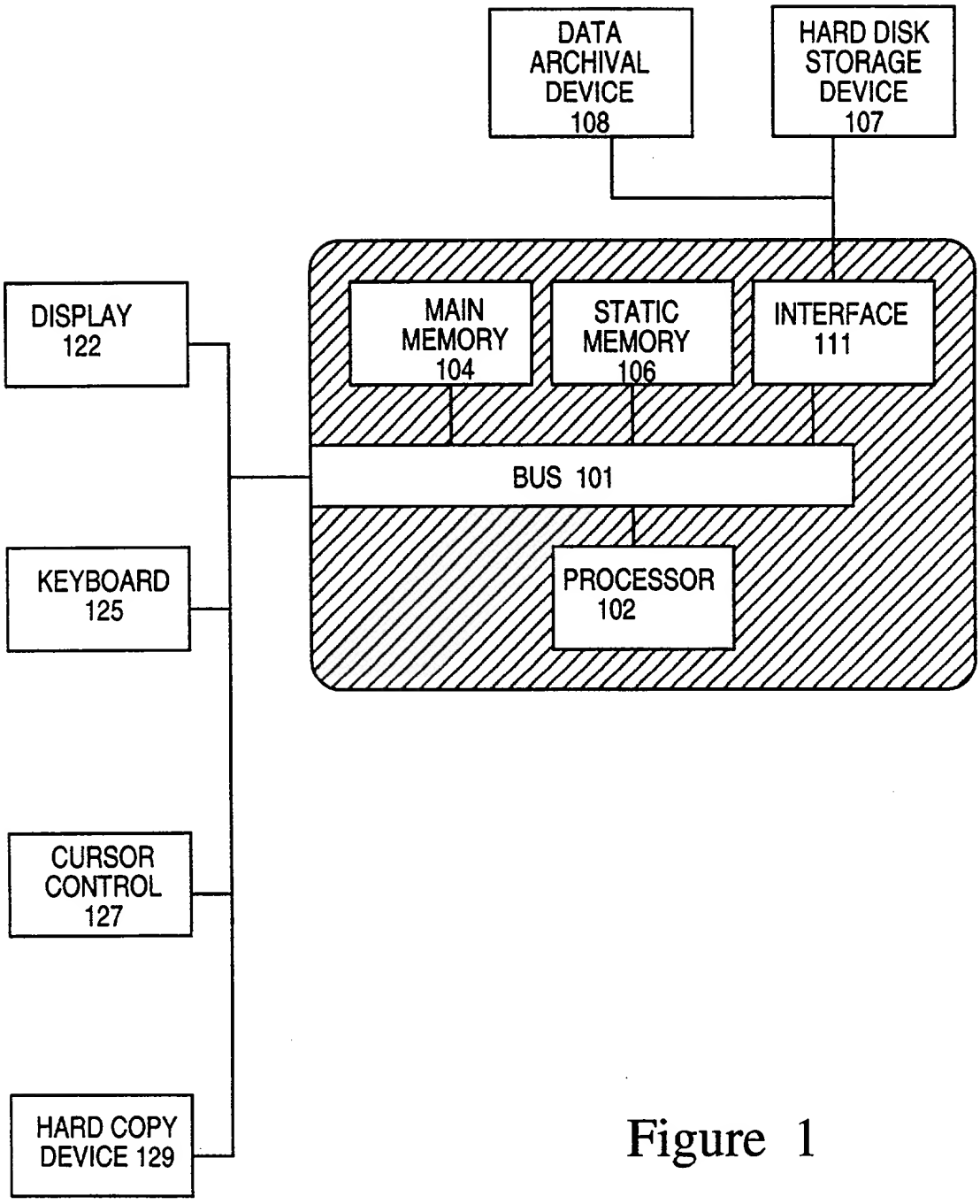
DATA ARCHIVAL DEVICE 108
HARD DISK STORAGE DEVICE 107
DISPLAY 122
MAIN MEMORY 104
STATIC MEMORY 106
INTERFACE 111
BUS 101
KEYBOARD 125
PROCESSOR 102
CURSOR CONTROL 127
HARD COPY DEVICE 129

Figure 1

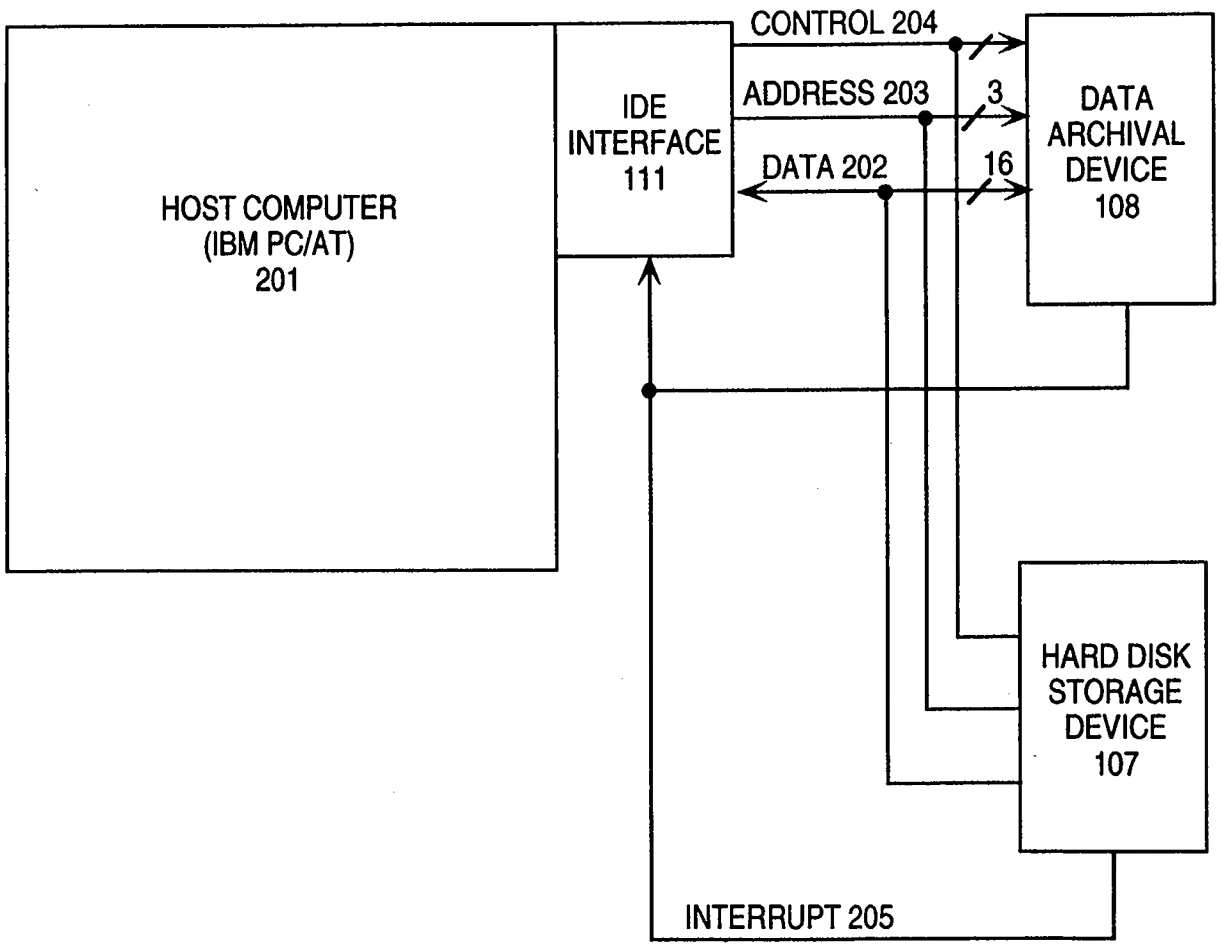
HOST COMPUTER
(IBM PC/AT)
201
IDE
INTERFACE
111
CONTROL 204
ADDRESS 203
3
DATA 202
16
DATA
ARCHIVAL
DEVICE
108
HARD DISK
STORAGE
DEVICE
107
INTERRUPT 205

Figure 2

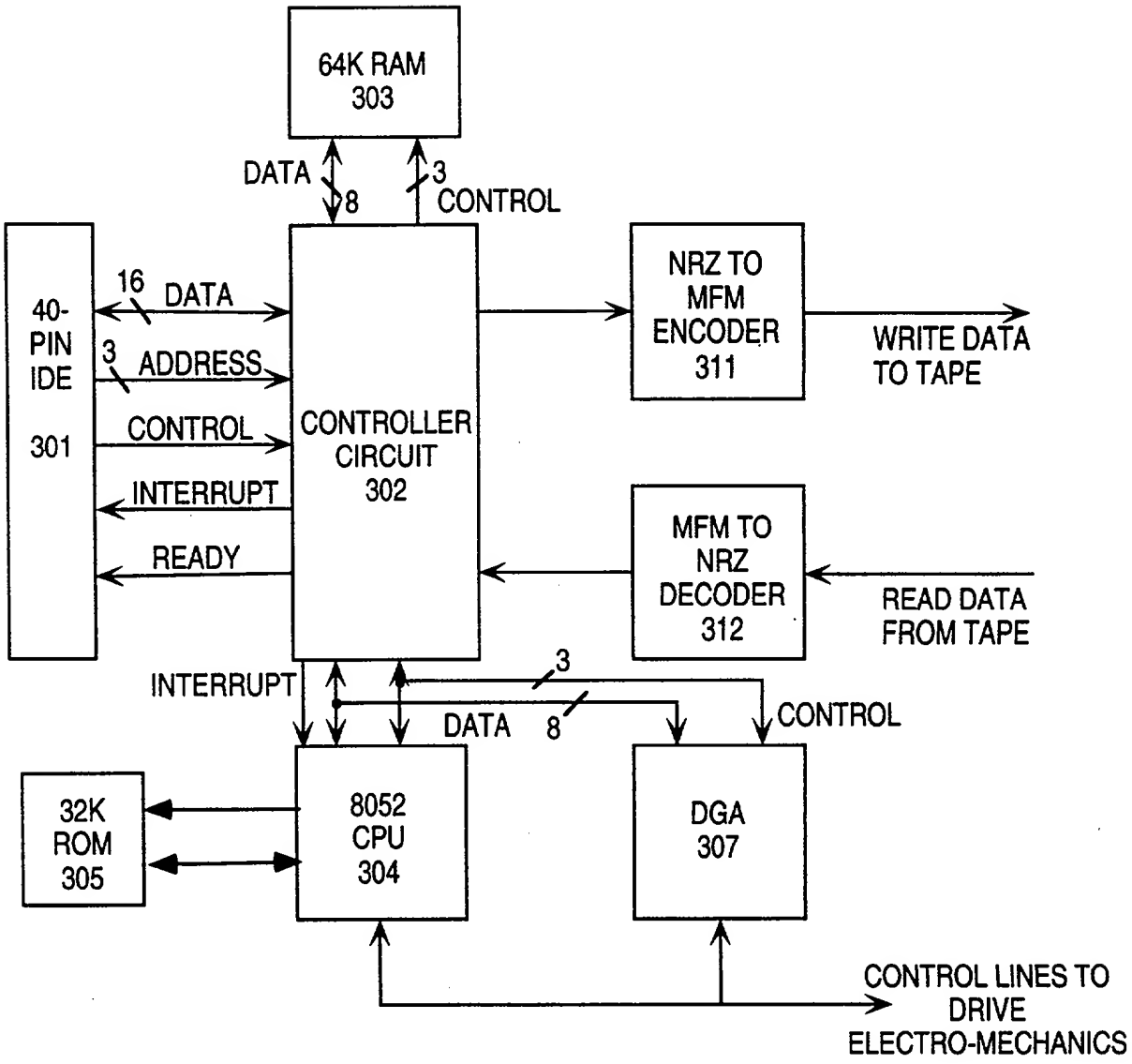
64K RAM
303
DATA
8
3
CONTROL
40-
PIN
IDE
301
16
DATA
3
ADDRESS
CONTROL
INTERRUPT
READY
CONTROLLER
CIRCUIT
302
NRZ TO
MFM
ENCODER
311
WRITE DATA
TO TAPE
MFM TO
NRZ
DECODER
312
READ DATA
FROM TAPE
INTERRUPT
3
DATA
8
CONTROL
32K
ROM
305
8052
CPU
304
DGA
307
CONTROL LINES TO
DRIVE
ELECTRO-MECHANICS

Figure 3

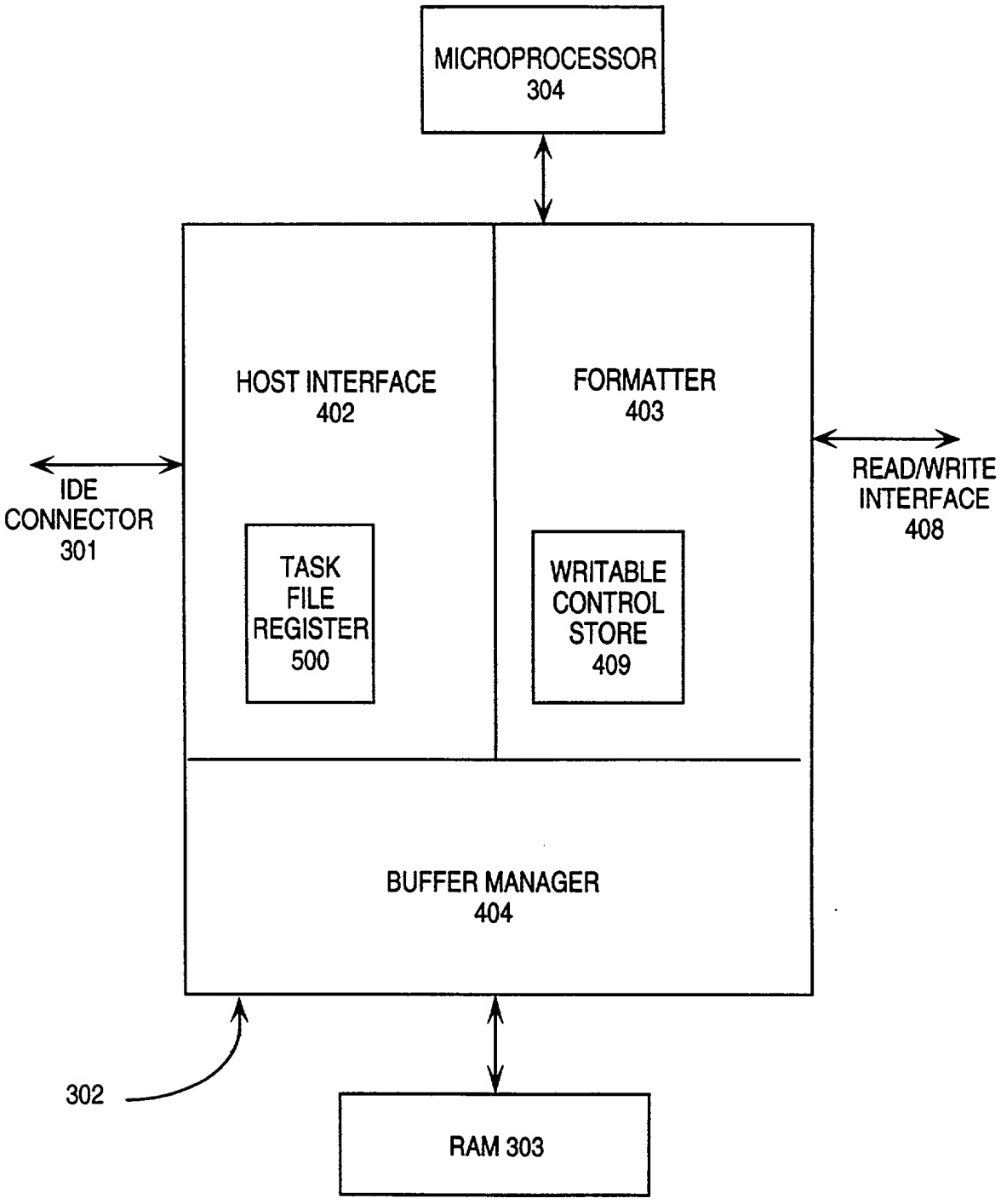
MICROPROCESSOR
304
HOST INTERFACE
402
FORMATTER
403
IDE
CONNECTOR
301
READ/WRITE
INTERFACE
408
TASK
FILE
REGISTER
500
WRITABLE
CONTROL
STORE
409
BUFFER MANAGER
404
302
RAM 303

Figure 4

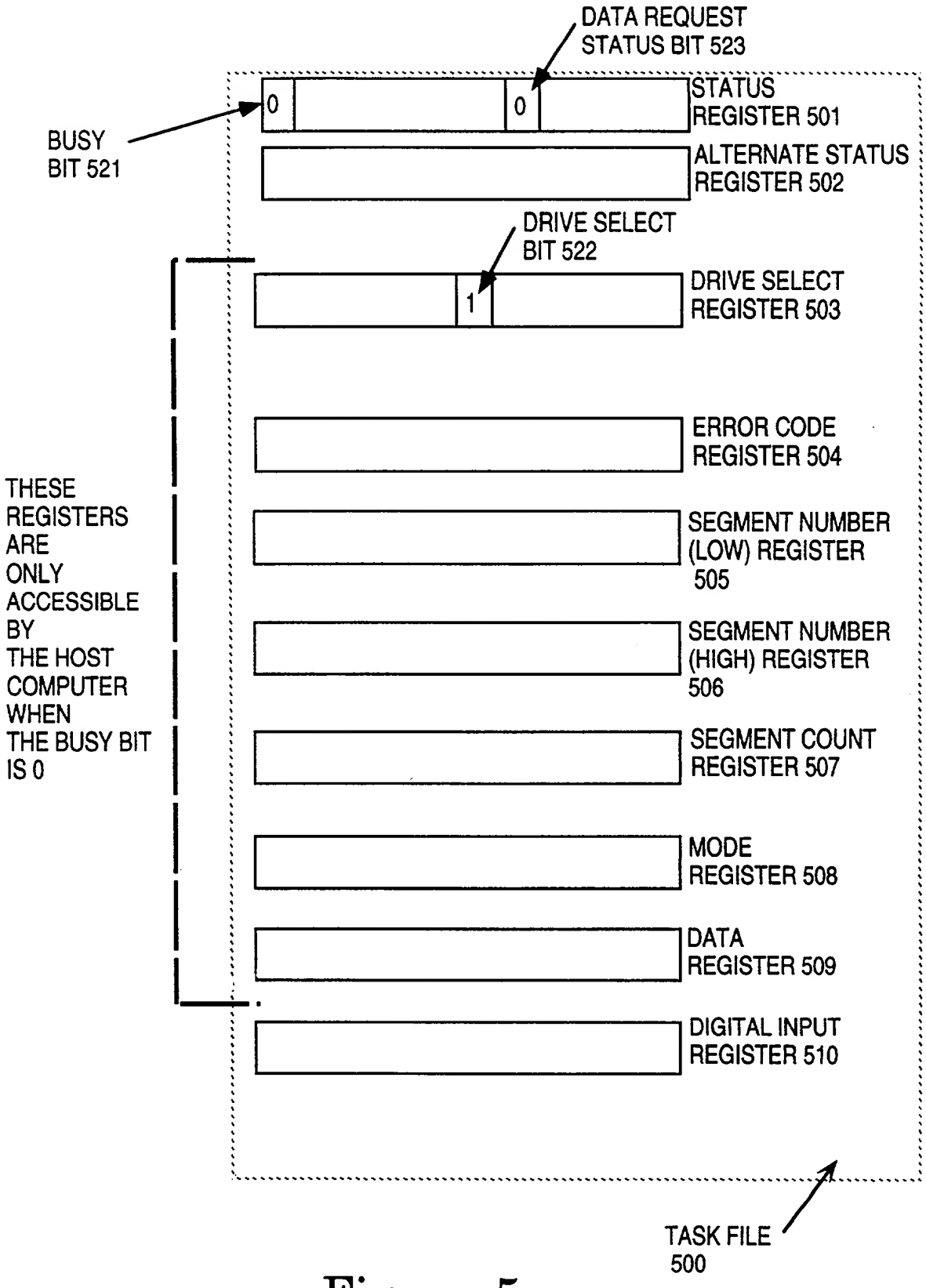
DATA REQUEST
STATUS BIT 523
0
0
STATUS
REGISTER 501
BUSY
BIT 521
ALTERNATE STATUS
REGISTER 502
DRIVE SELECT
BIT 522
1
DRIVE SELECT
REGISTER 503
ERROR CODE
REGISTER 504
THESE
REGISTERS
ARE
ONLY
ACCESSIBLE
BY
THE HOST
COMPUTER
WHEN
THE BUSY BIT
IS 0
SEGMENT NUMBER
(LOW) REGISTER
505
SEGMENT NUMBER
(HIGH) REGISTER
506
SEGMENT COUNT
REGISTER 507
MODE
REGISTER 508
DATA
REGISTER 509
DIGITAL INPUT
REGISTER 510
TASK FILE
500

Figure 5

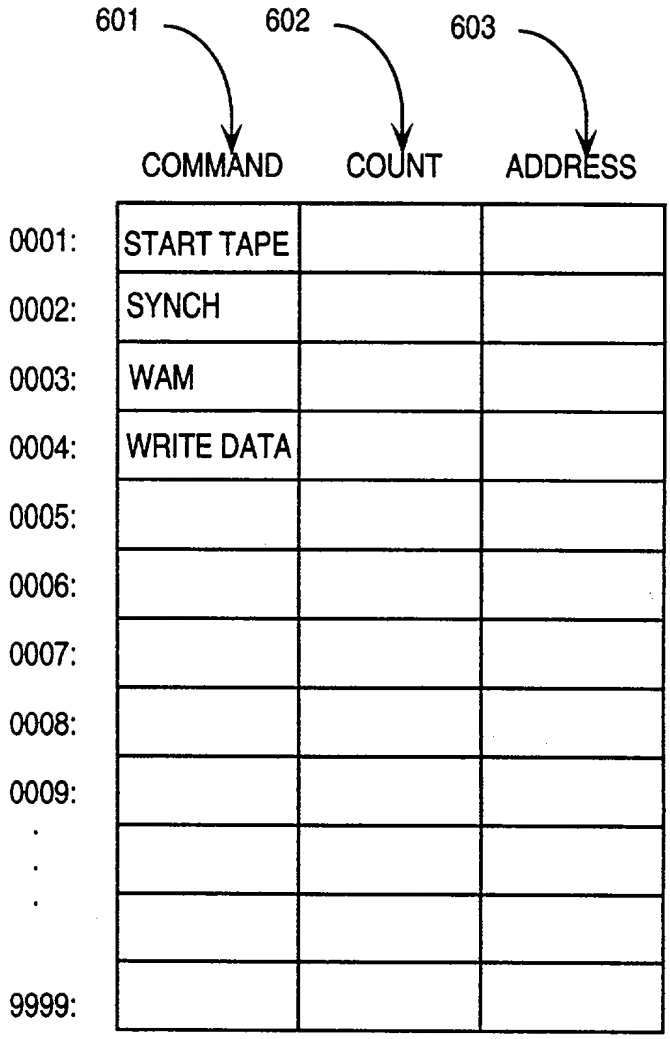
601
602
603
COMMAND
COUNT
ADDRESS
0001:
START TAPE
0002:
SYNCH
0003:
WAM
0004:
WRITE DATA
0005:
0006:
0007:
0008:
0009:
9999:

Figure 6

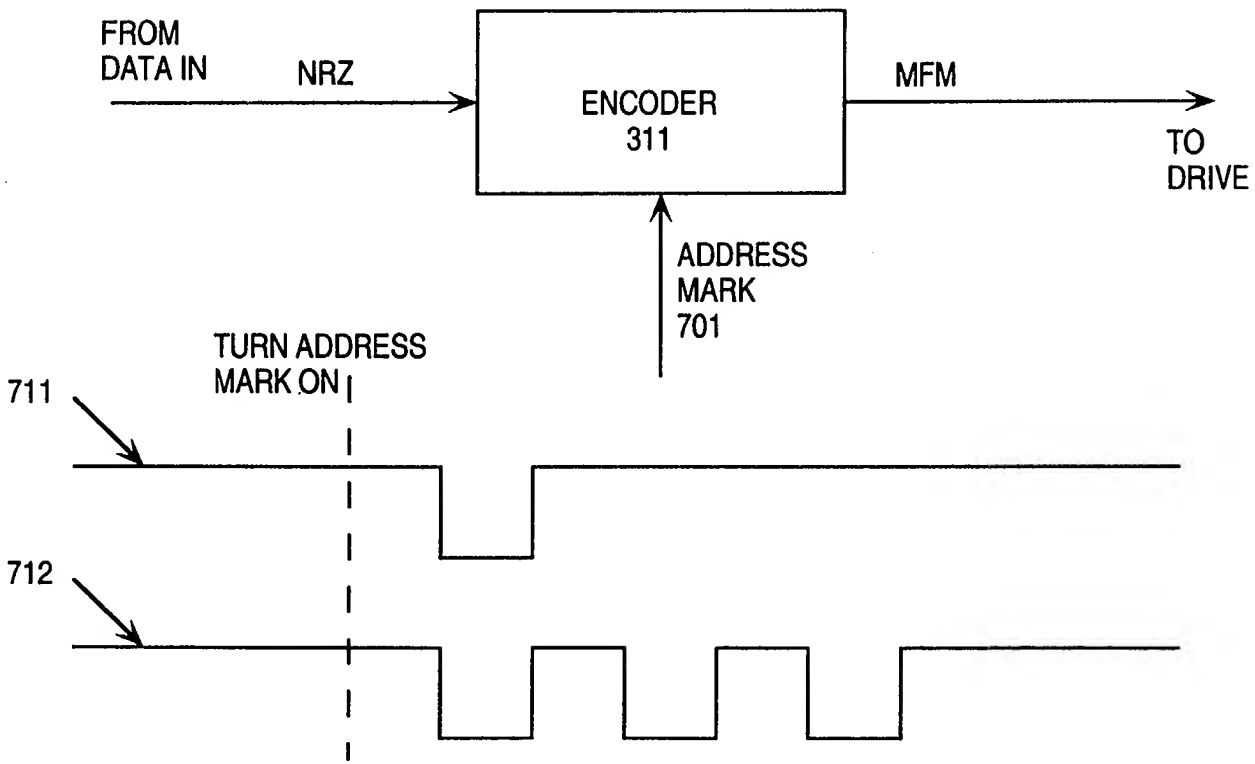
FROM
DATA IN
NRZ
ENCODER
311
MFM
TO
DRIVE
ADDRESS
MARK
701
TURN ADDRESS
MARK ON
711
712

Figure 7(a)

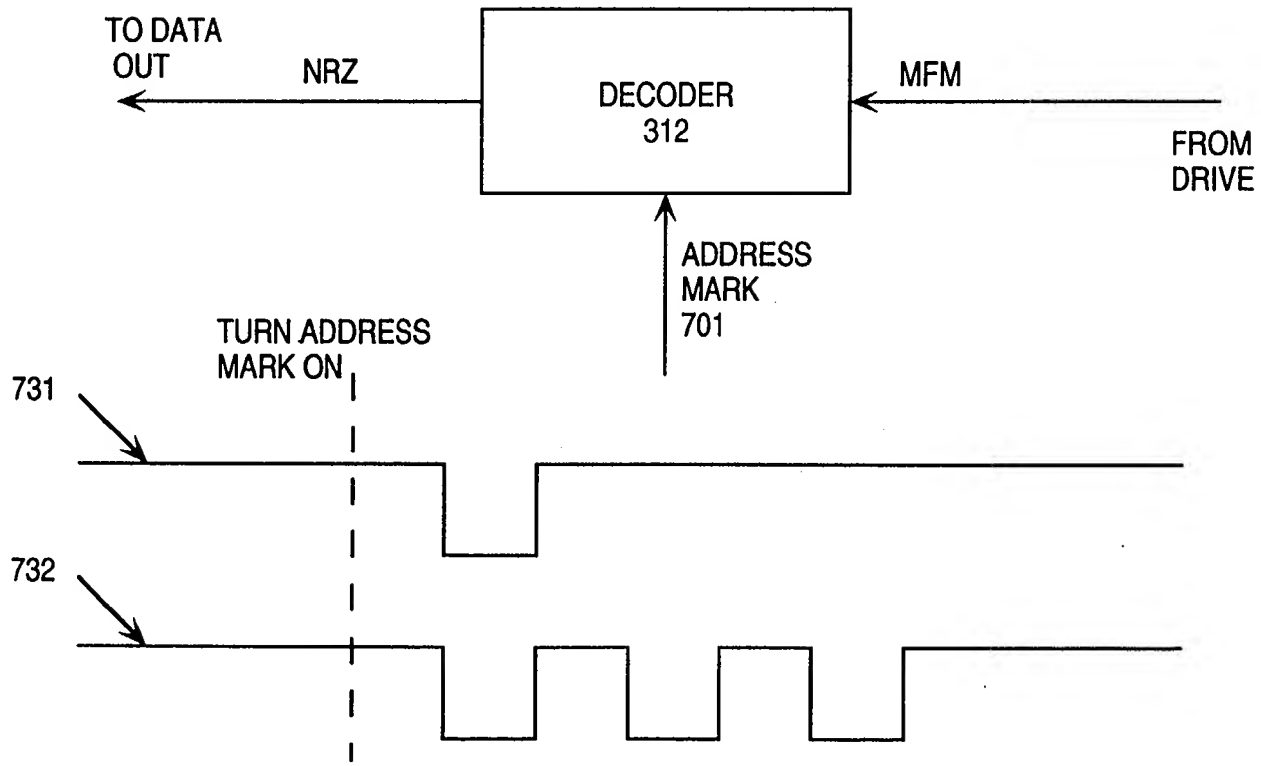
TO DATA
OUT
NRZ
DECODER
312
MFM
FROM
DRIVE
ADDRESS
MARK
701
TURN ADDRESS
MARK ON
731
732

Figure 7(b)

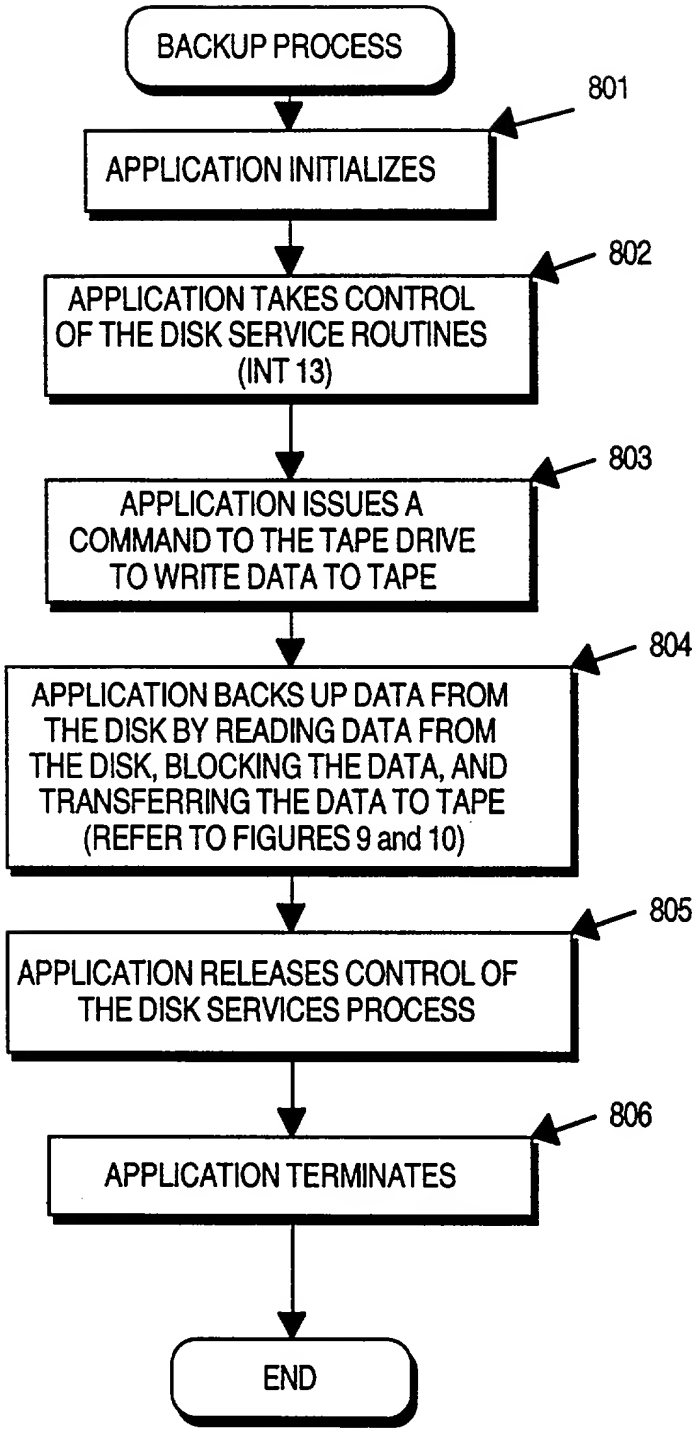
BACKUP PROCESS
801
APPLICATION INITIALIZES
802
APPLICATION TAKES CONTROL OF THE DISK SERVICE ROUTINES (INT 13)
803
APPLICATION ISSUES A COMMAND TO THE TAPE DRIVE TO WRITE DATA TO TAPE
804
APPLICATION BACKS UP DATA FROM THE DISK BY READING DATA FROM THE DISK, BLOCKING THE DATA, AND TRANSFERRING THE DATA TO TAPE (REFER TO FIGURES 9 and 10)
805
APPLICATION RELEASES CONTROL OF THE DISK SERVICES PROCESS
806
APPLICATION TERMINATES
END

Figure 8

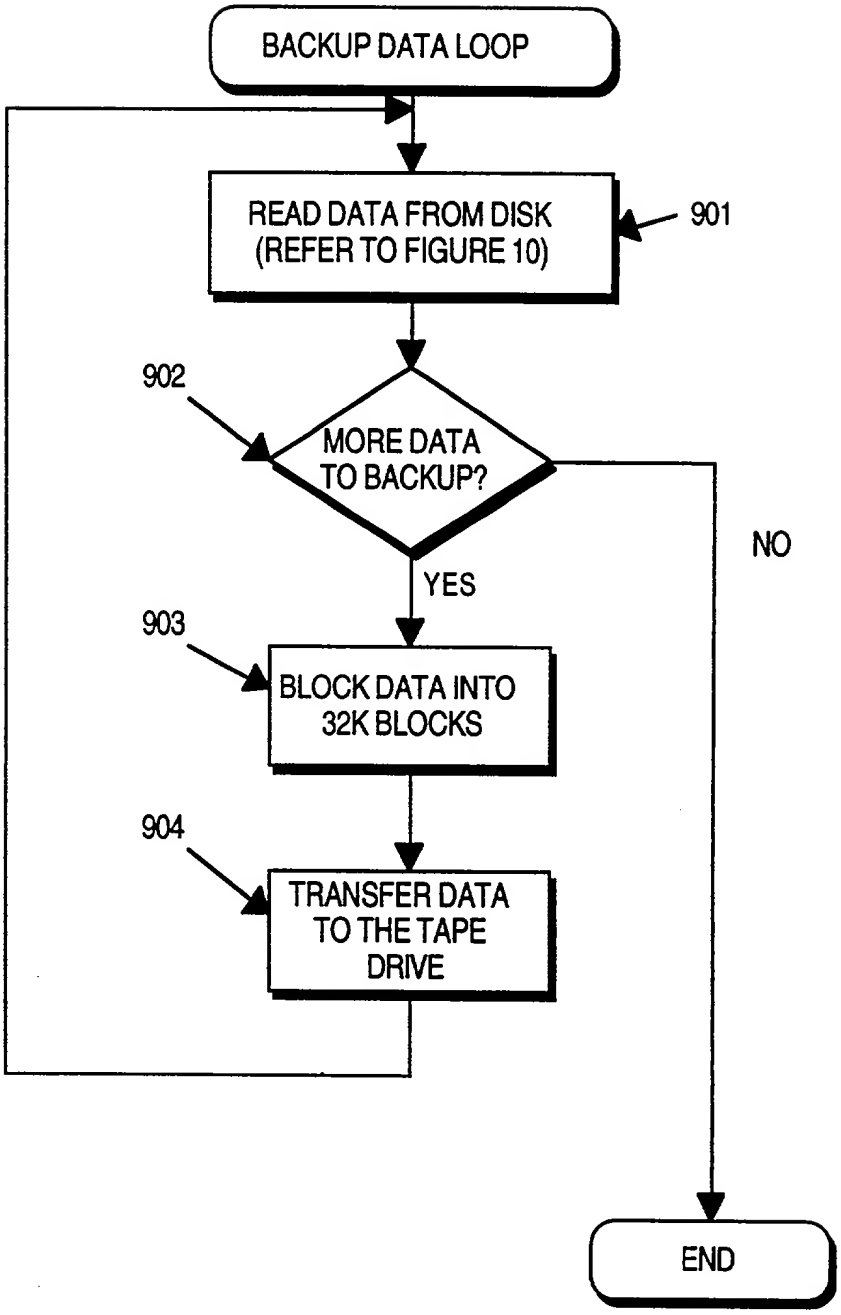
BACKUP DATA LOOP
READ DATA FROM DISK
(REFER TO FIGURE 10)
901
902
MORE DATA
TO BACKUP?
NO
YES
903
BLOCK DATA INTO
32K BLOCKS
904
TRANSFER DATA
TO THE TAPE
DRIVE
END

Figure 9

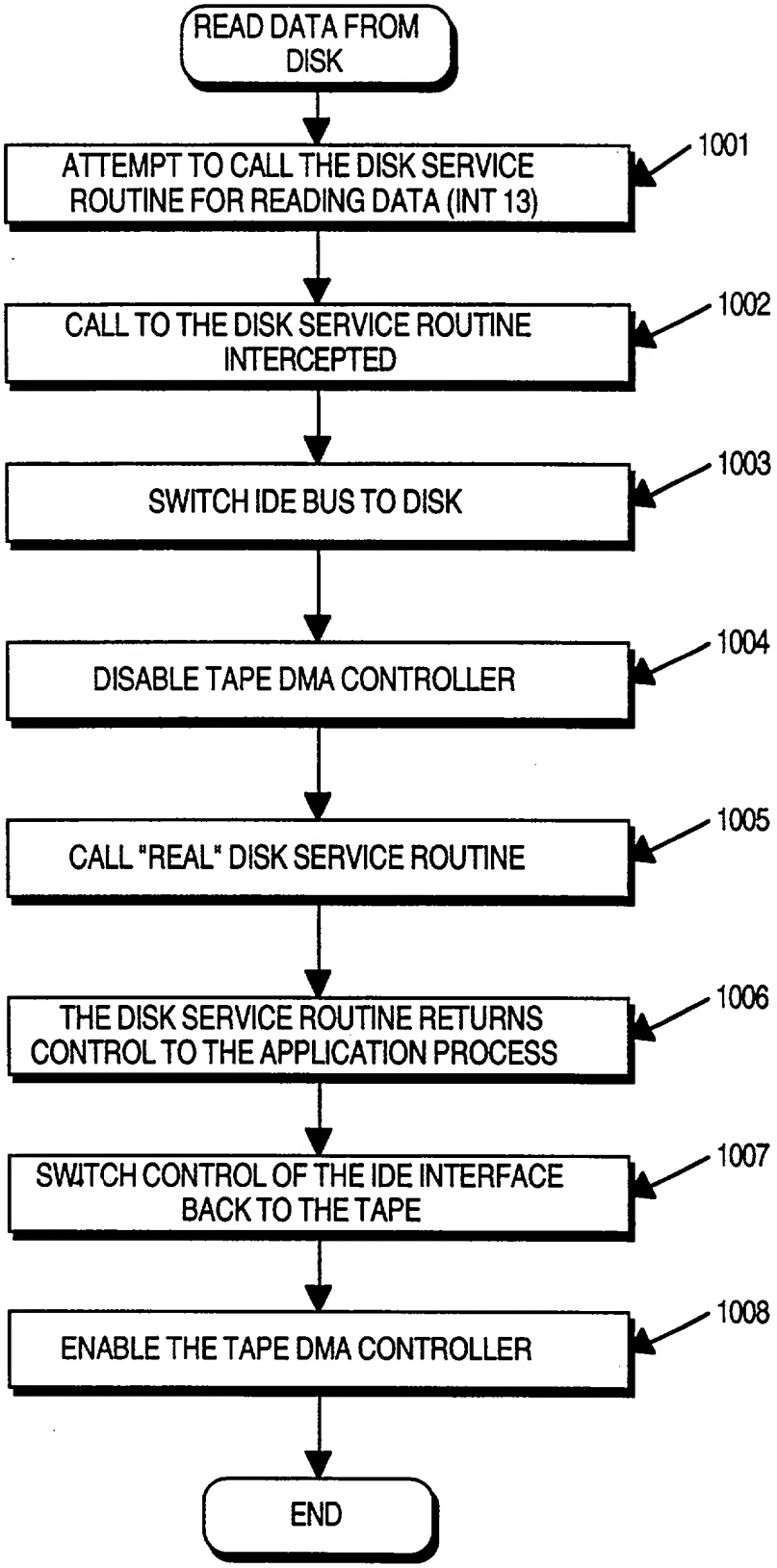
READ DATA FROM DISK
ATTEMPT TO CALL THE DISK SERVICE ROUTINE FOR READING DATA (INT 13)
1001
CALL TO THE DISK SERVICE ROUTINE INTERCEPTED
1002
SWITCH IDE BUS TO DISK
1003
DISABLE TAPE DMA CONTROLLER
1004
CALL "REAL" DISK SERVICE ROUTINE
1005
THE DISK SERVICE ROUTINE RETURNS CONTROL TO THE APPLICATION PROCESS
1006
SWITCH CONTROL OF THE IDE INTERFACE BACK TO THE TAPE
1007
ENABLE THE TAPE DMA CONTROLLER
1008
END

Figure 10

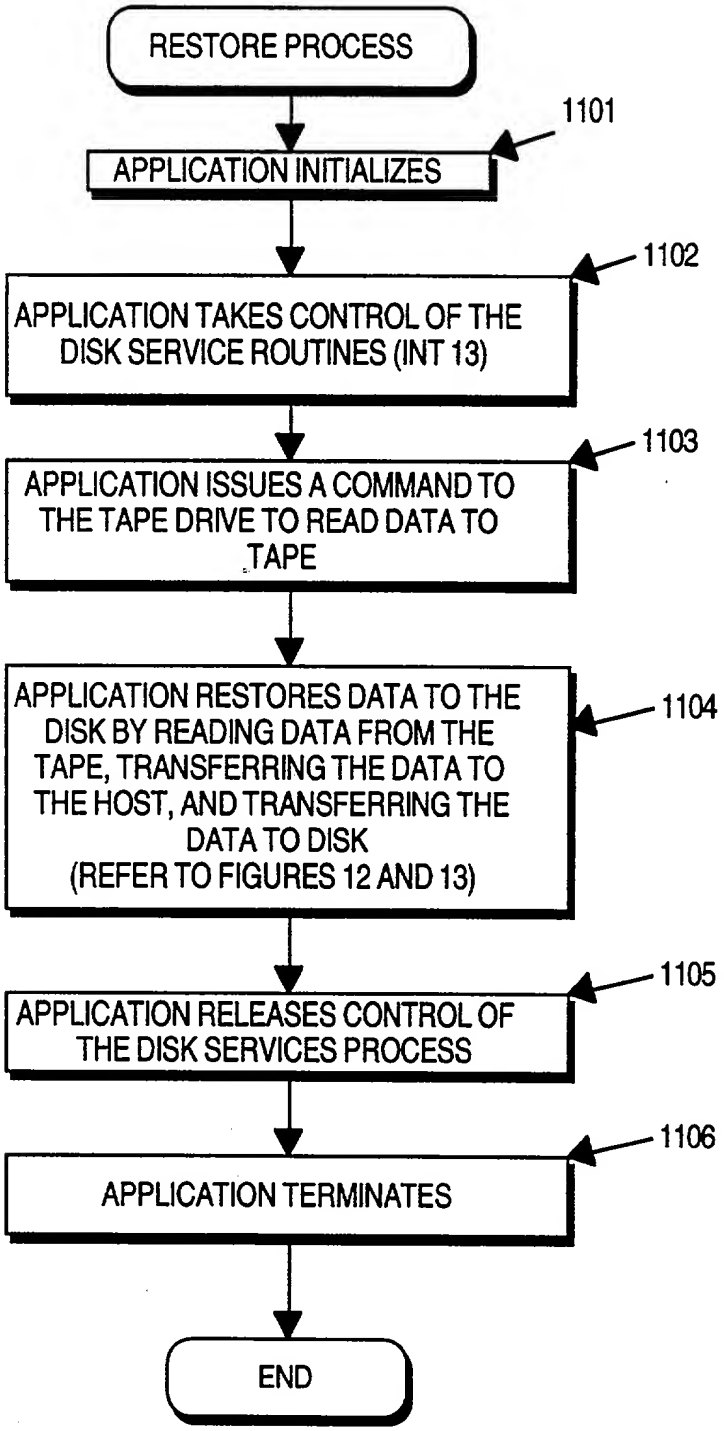
RESTORE PROCESS
1101
APPLICATION INITIALIZES
1102
APPLICATION TAKES CONTROL OF THE DISK SERVICE ROUTINES (INT 13)
1103
APPLICATION ISSUES A COMMAND TO THE TAPE DRIVE TO READ DATA TO TAPE
APPLICATION RESTORES DATA TO THE DISK BY READING DATA FROM THE TAPE, TRANSFERRING THE DATA TO THE HOST, AND TRANSFERRING THE DATA TO DISK (REFER TO FIGURES 12 AND 13)
1104
1105
APPLICATION RELEASES CONTROL OF THE DISK SERVICES PROCESS
1106
APPLICATION TERMINATES
END

Figure 11

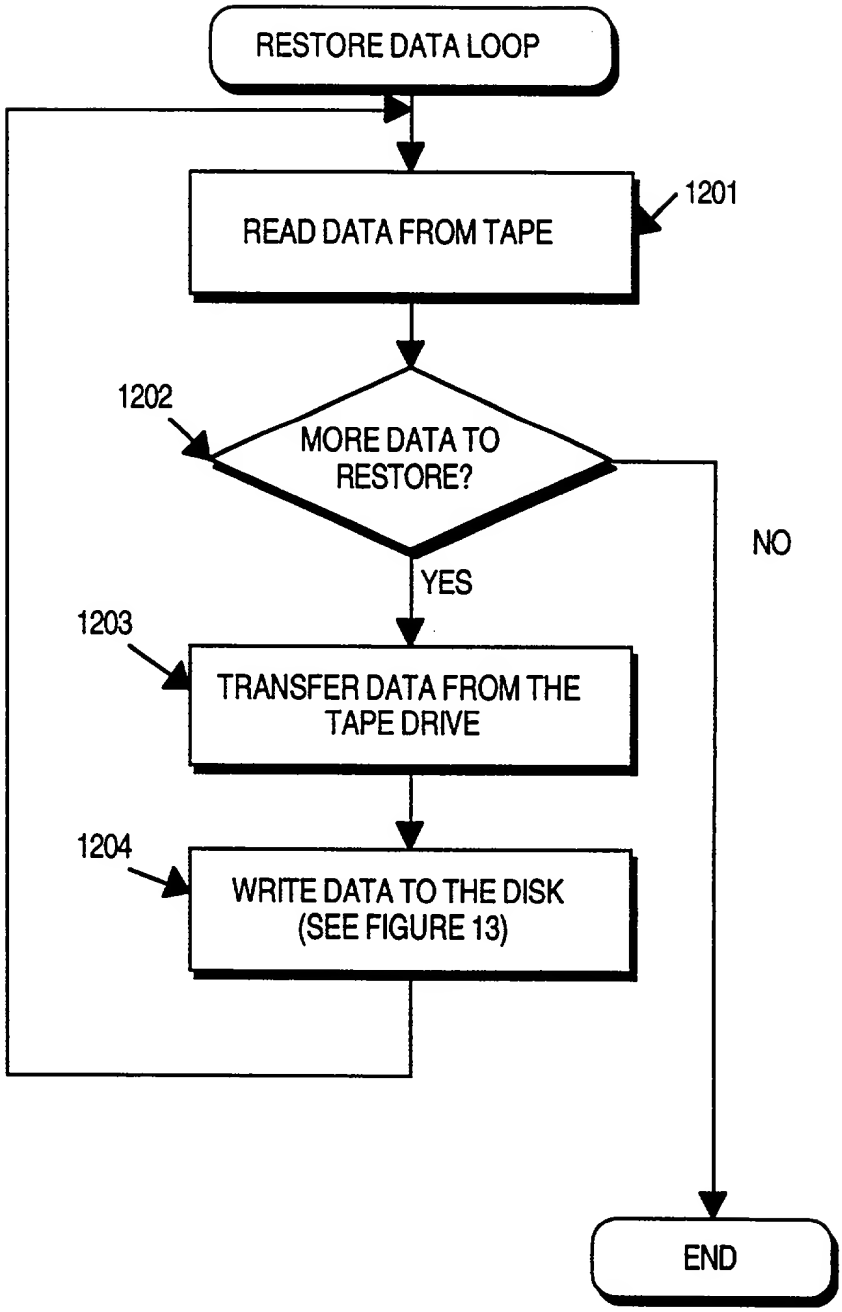
RESTORE DATA LOOP
READ DATA FROM TAPE
1201
1202
MORE DATA TO RESTORE?
NO
YES
1203
TRANSFER DATA FROM THE TAPE DRIVE
1204
WRITE DATA TO THE DISK (SEE FIGURE 13)
END

Figure 12

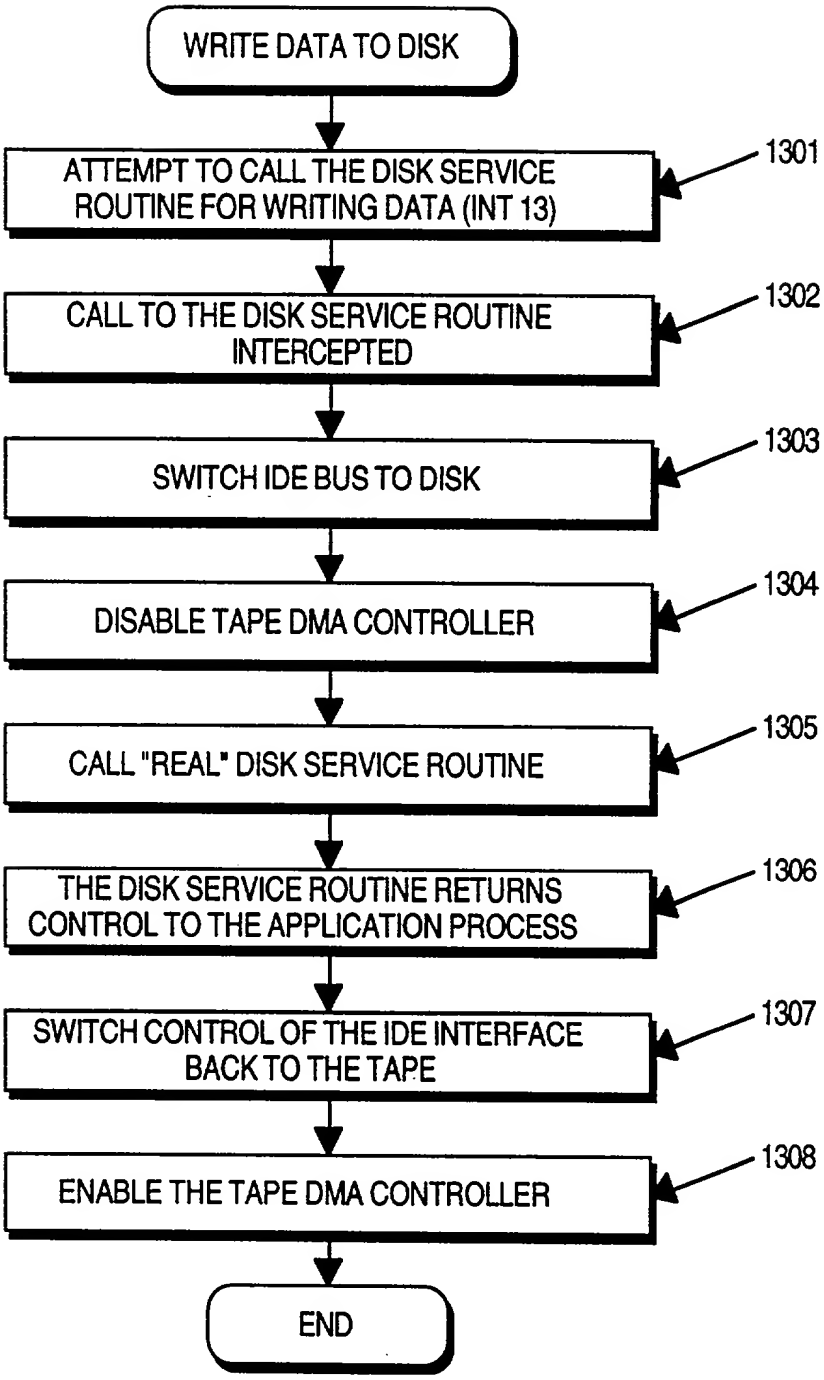
WRITE DATA TO DISK
ATTEMPT TO CALL THE DISK SERVICE ROUTINE FOR WRITING DATA (INT 13)
1301
CALL TO THE DISK SERVICE ROUTINE INTERCEPTED
1302
SWITCH IDE BUS TO DISK
1303
DISABLE TAPE DMA CONTROLLER
1304
CALL "REAL" DISK SERVICE ROUTINE
1305
THE DISK SERVICE ROUTINE RETURNS CONTROL TO THE APPLICATION PROCESS
1306
SWITCH CONTROL OF THE IDE INTERFACE BACK TO THE TAPE
1307
ENABLE THE TAPE DMA CONTROLLER
1308
END

Figure 13

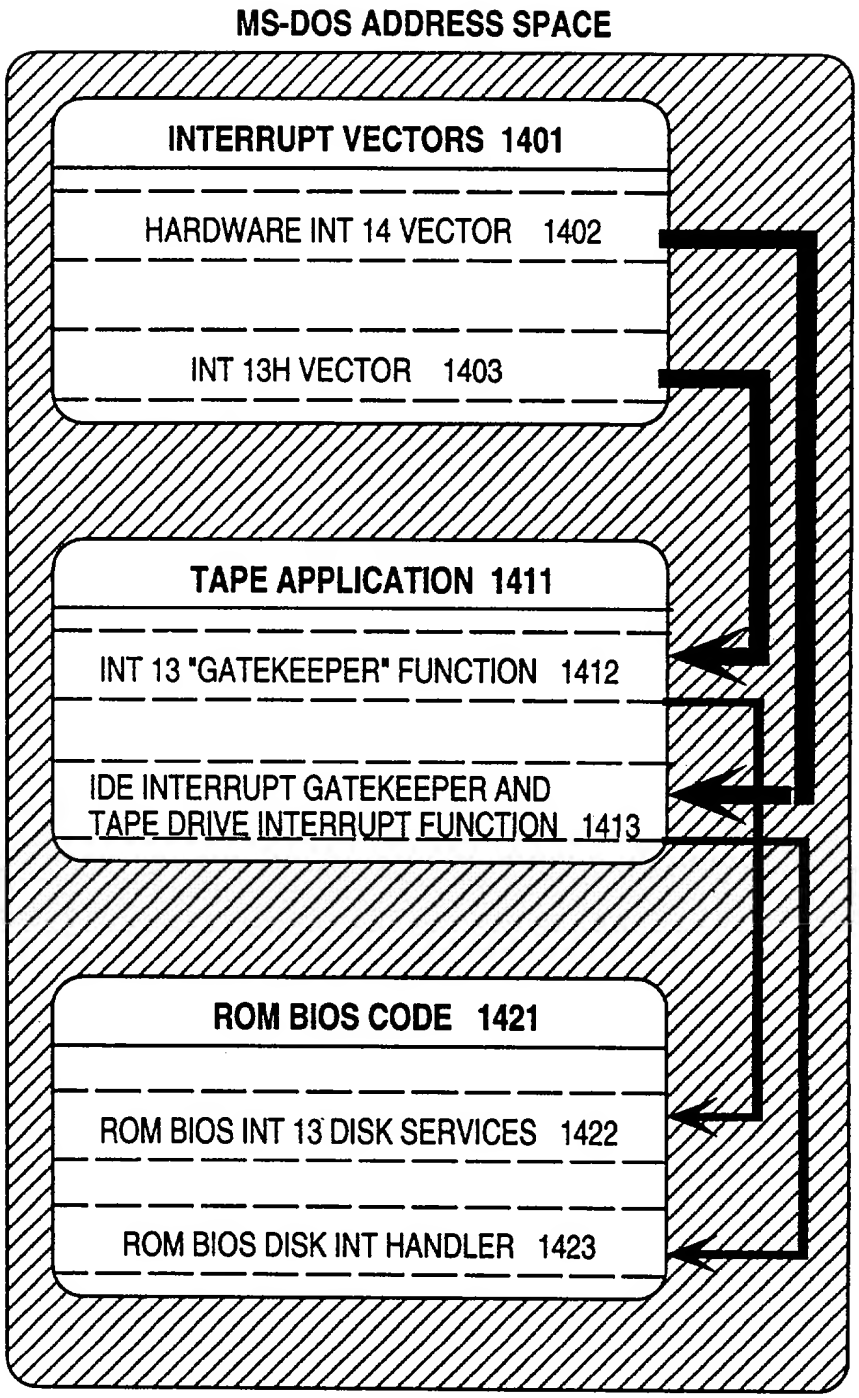
MS-DOS ADDRESS SPACE
INTERRUPT VECTORS 1401
HARDWARE INT 14 VECTOR 1402
INT 13H VECTOR 1403
TAPE APPLICATION 1411
INT 13 "GATEKEEPER" FUNCTION 1412
IDE INTERRUPT GATEKEEPER AND
TAPE DRIVE INTERRUPT FUNCTION 1413
ROM BIOS CODE 1421
ROM BIOS INT 13 DISK SERVICES 1422
ROM BIOS DISK INT HANDLER 1423

Figure 14

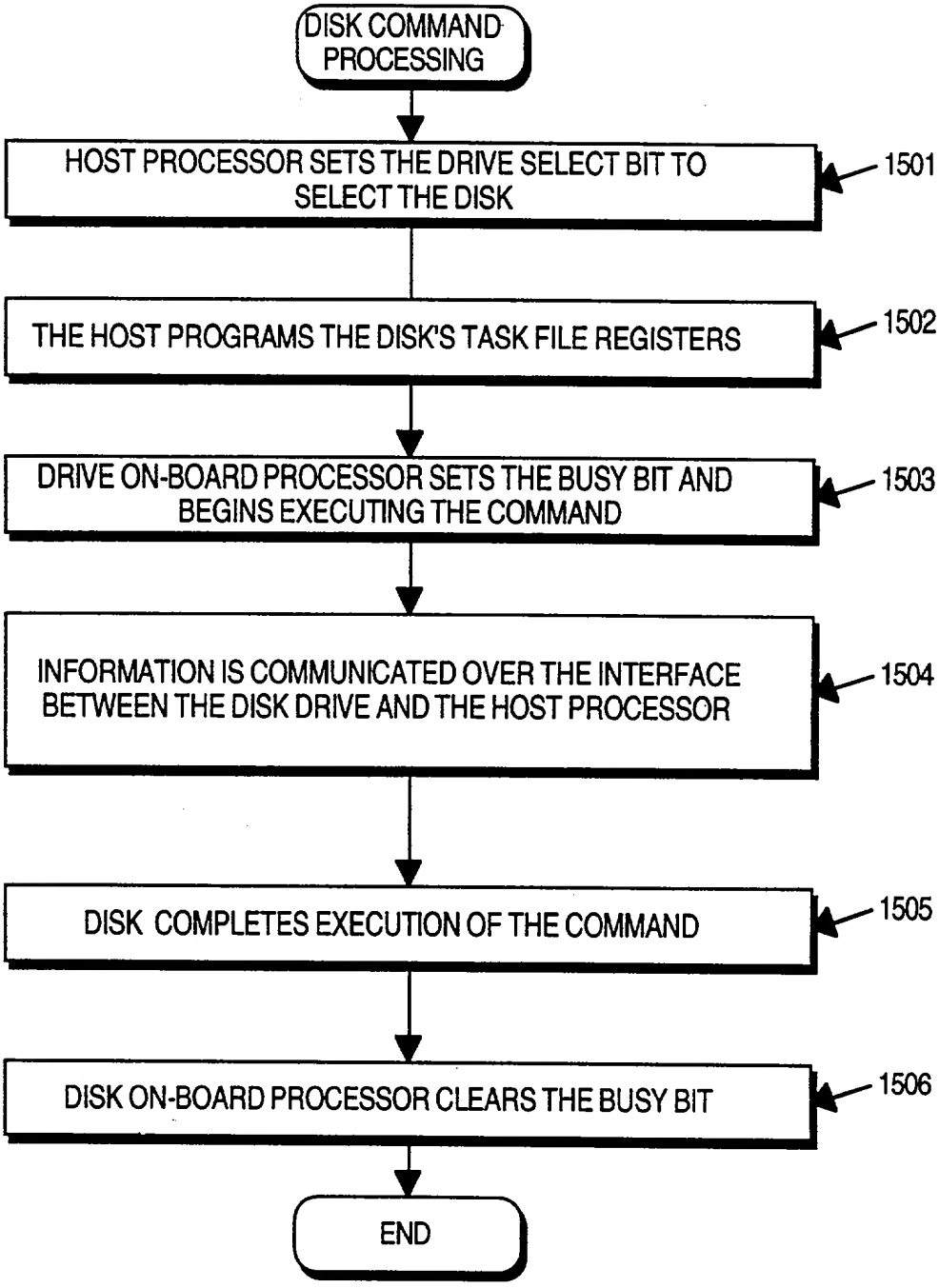

Figure 15
(Prior Art)

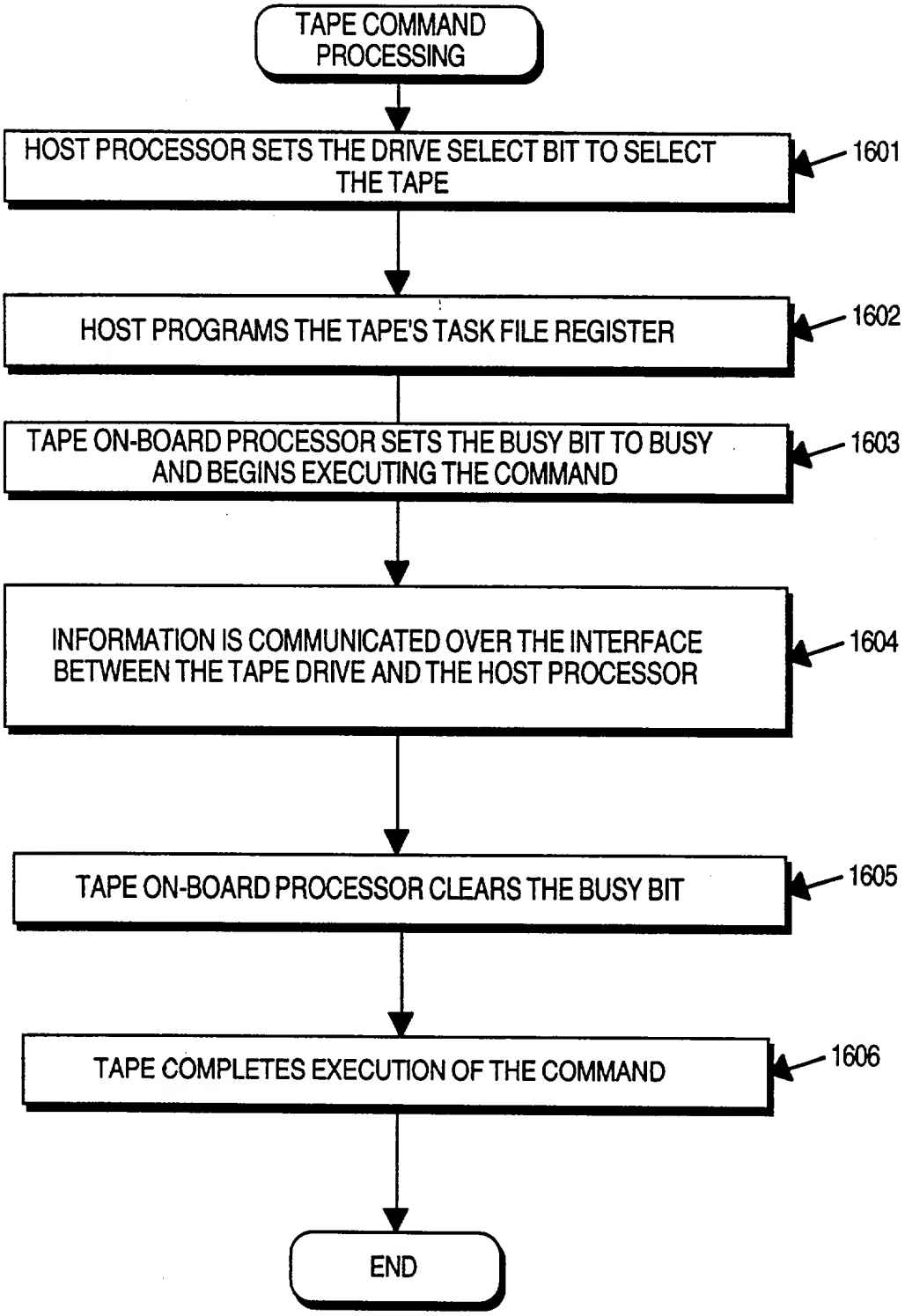
TAPE COMMAND PROCESSING
HOST PROCESSOR SETS THE DRIVE SELECT BIT TO SELECT THE TAPE
1601
HOST PROGRAMS THE TAPE'S TASK FILE REGISTER
1602
TAPE ON-BOARD PROCESSOR SETS THE BUSY BIT TO BUSY AND BEGINS EXECUTING THE COMMAND
1603
INFORMATION IS COMMUNICATED OVER THE INTERFACE BETWEEN THE TAPE DRIVE AND THE HOST PROCESSOR
1604
TAPE ON-BOARD PROCESSOR CLEARS THE BUSY BIT
1605
TAPE COMPLETES EXECUTION OF THE COMMAND
1606
END

Figure 16

DECLARATION AND POWER OF ATTORNEY FOR PATENT APPLICATION

As a below named inventor, I hereby declare that:

My residence, post office address and citizenship are as stated below, next to my name.

I believe I am the original, first, and sole inventor (if only one name is listed below) or an original, first, and joint inventor (if plural names are listed below) of the subject matter which is claimed and for which a patent is sought on the invention entitled

TAPE BACKUP SYSTEM

the specification of which

____ is attached hereto.
XX was filed on 11/13/90 as
Application Serial No. 07/612,540
and was amended on _____.
(if applicable)

I hereby state that I have reviewed and understand the contents of the above-identified specification, including the claims, as amended by any amendment referred to above. I do not know and do not believe that the same was ever known or used in the United States of America before my invention thereof, or patented or described in any printed publication in any country before my invention thereof or more than one year prior to this application, that the same was not in public use or on sale in the United States of America more than one year prior to this application, and that the invention has not been patented or made the subject of an inventor's certificate issued before the date of this application in any country foreign to the United States of America on an application filed by me or my legal representatives or assigns more than twelve months prior to this application.

I acknowledge the duty to disclose information which is material to the examination of this application in accordance with Title 37, Code of Federal Regulations, Section 1.56(a).

I hereby claim foreign priority benefits under Title 35, United States Code, Section 119, of any foreign application(s) for patent or inventor's certificate listed below and have also identified below any foreign application for patent or inventor's certificate having a filing date before that of the application on which priority is claimed:

Prior Foreign Application(s)			Priority Claimed	
(Number)	(Country)	(Day/Month/Year Filed)	Yes	No
(Number)	(Country)	(Day/Month/Year Filed)	Yes	No
(Number)	(Country)	(Day/Month/Year Filed)	Yes	No

I hereby claim the benefit under Title 35, United States Code, Section 120 of any United States application(s) listed below and, insofar as the subject matter of each of the claims of this application is not disclosed in the prior United States application in the manner provided by the first paragraph of Title 35, United States Code, Section 112, I acknowledge the duty to disclose material information as defined in Title 37, Code of Federal Regulations, Section 1.56(a) which occurred between the filing date of the prior application and the national or PCT international filing date of this application:

(Application Serial No.)	Filing Date	(Status -- patented, pending, abandoned)
(Application Serial No.)	Filing Date	(Status -- patented, pending, abandoned)

25,831; Jeffrey Jay Blatt, Reg. No. 30,244; Stephen D. Gross, Reg. No. 31,020; David R. Halvorson, Reg. No. 33,395; George W. Hoover, Reg. No. 32,992; Michael Hurey, Reg. No. 33,513; Eric S. Hyman, Reg. No. 30,139; Stephen L. King, Reg. No. 19,180; Maria E. McCormack, Reg. No. 31,639; Ronald W. Reagin, Reg. No. 20,340; James C. Scheller, Reg. No. 31,195; Ira M. Siegel, Reg. No. 28,907; Stanley W. Sokoloff, Reg. No. 25,128; Edwin H. Taylor, Reg. No. 25,129; Lester J. Vincent, Reg. No. 31,460; and Norman Zafman, Reg. No. 26,250; my attorneys and Vernon Randall Gard, Reg. No. 33,886; and Keith G. Askoff, Reg. No. 33,828, my patent agents; of BLAKELY, SOKOLOFF, TAYLOR & ZAFMAN, with offices located at 12400 Wilshire Boulevard, 7th Floor, Los Angeles, California 90025, telephone (213) 207-3800, with full power of substitution and revocation, to prosecute this application and to transact all business in the Patent and Trademark Office connected herewith.

I hereby declare that all statements made herein of my own knowledge are true and that all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such wilful false statements may jeopardize the validity of the application or any patent issued thereon.

Full Name of Sole/First Inventor FAAN-HOAN LIU

Inventor's Signature Faan Hoan Liu Date January 10, 1991

Residence Santa Cruz, California (City, State) Citizenship U.S.A. (Country)

Post Office Address 1600 Thompson Ave
Santa Cruz, CA 95062

Full Name of Second/Joint Inventor JORGE GUSTAVSON

Inventor's Signature Jorge Gustavson Date Jan 10, 1991

Residence ~~506~~ Santa Cruz CA (City, State) Citizenship U.S.A. (Country)

Post Office Address 506 Murray St
Santa Cruz CA 95062-3736

Full Name of Third/Joint Inventor _____

Inventor's Signature _____ Date _____

Residence _____ (City, State) Citizenship _____ (Country)

Post Office Address _____

Full Name of Fourth/Joint Inventor _____

Inventor's Signature _____ Date _____

Residence _____ (City, State) Citizenship _____ (Country)

Post Office Address _____